



ScuDo
Scuola di Dottorato - Doctoral School
WHAT YOU ARE. TAKES YOU FAR



**UNIVERSITÀ
DEGLI STUDI
DI TORINO**

Doctoral Dissertation

Doctoral Program in Pure and Applied Mathematics (30th cycle)

Data assimilation for neurobiology modelling

Nicola Politi

Supervisors:

Prof. Claudio Canuto, Politecnico di Torino

Prof. Wenlian Lu, Fudan University 复旦大学

Doctoral Examination Committee:

Prof. Davide Carlo Ambrosi, Politecnico di Torino

Prof. Susanne Ditlevsen, Referee, University of Copenhagen

Prof. Giovanni Naldi, Referee, Università degli studi di Milano

Prof. Sandra Pieraccini, Politecnico di Torino

Prof. Gianluigi Rozza, SISSA

Politecnico di Torino
10 September 2018

Acknowledgements

First, I wish to thank my supervisors Claudio Canuto and Wenlian Lu as well as Jianfeng Feng for guiding me through my research work and the referees Susanne Ditlevsen and Giovanni Naldi for their stimulating advice. Then, my special acknowledgement go to Silvia Jimenez, Samuel Kerrien, Christian Rössert, and Werner Van Geit for the invaluable help in providing me and answer my questions on the BBP experimental dataset, to Henry Abarbanel, Paul Rozdeba, and Jingxin Ye for their support on the use of minAone, and to Enrico Bibbona, Stephen Coombes, Angelo Di Garbo, Mauro Gasparini, Pedro Machado, Gianluca Mastrantonio, and Wenbo Sheng for the suggestions and the precious counsel. Finally, I want to show my great appreciation to Alessandro, Annachiara, Caterina, Davide, Filippo, Giulia, Deng, Lorenzo, Stefano, Wei, Yunyi, and everyone who shared and bore with me during the last four years.

Contents

Acknowledgements	iii
Introduction	1
I Data assimilation methods	7
Introduction to Part I	9
1 Generalities on state-space models	13
1.1 The state-space model framework	13
2 Off-line smoothing methods	21
2.1 The Markov chain Monte Carlo methodology	26
2.1.1 The Metropolis-Hastings framework	29
2.2 Variational methods	33
2.2.1 The path integral framework	36
2.2.2 minAone	38
3 On-line filtering methods	43
3.1 Approximate Gaussian filters	45
3.1.1 The Kalman filter	45
3.1.2 The extended and the unscented Kalman filter	47
3.1.3 The ensemble Kalman filter	48
3.2 The particle filter	50
3.2.1 The bootstrap filter	51
3.2.2 The optimal sequential importance resampling	52
3.2.3 Gaussian particle filters	54
3.2.4 Well-posedness and consistency of particle filters	54

4	Practical issues arising in data assimilation	57
4.1	Handling parameters	58
4.1.1	Data assimilation for parameter estimation	58
4.1.2	State-space model for hyperparameters estimation	62
4.2	Twin experiments from continuous-time models	66
4.3	Dealing with bounded variables	70
4.3.1	Single inequality constraint	73
4.3.2	Bounded-interval constraint	74
4.4	Evaluating data assimilation methods	76
4.4.1	Performance score for algorithm evaluation	79
II	Neurobiology modelling	83
	Introduction to Part II	85
5	Modelling of single neurons	87
5.1	Generalities on single neuron models	87
5.1.1	Ionic currents and state variables	89
5.2	Toy model for a single neuron	96
5.2.1	Sample toy model trajectory	98
5.3	Realistic model from the Blue Brain Project	100
5.3.1	Mathematical model for L23_PC_cADpyr229.1	101
5.3.2	Model parameters	104
5.3.3	Sample dynamics and artificial dataset	104
5.4	Mathematical properties of single-neuron models	107
5.4.1	Well-posedness of ODE single-neuron models with constant input current	110
5.4.2	Qualitative analysis and numerical aspects in neuronal modelling	111
5.4.3	More on well-posedness of single-neuron models	114
6	Neural network models	119
6.1	Neocortical microcircuit composition	119
6.2	Network model of leaky integrate and fire neurons	120
6.2.1	Synaptic current modelling	124
6.2.2	Network parameters	126
6.2.3	Neural population activity measure	127
6.2.4	Sample network time course	127

7	Spike train metrics	131
7.1	Cost-based spike metrics	132
7.1.1	Victor-Purpura SPIKE distance	132
7.1.2	Victor-Purpura ISI distance	133
7.2	Embedding-based spike metrics	134
7.2.1	Van Rossum distance	134
7.2.2	Similarity measures	135
7.3	Parameter-free spike metrics	136
7.3.1	ISI- and SPIKE-distance	136
7.3.2	SPIKE synchronization	139
7.4	Population extensions	140
III	Results on data assimilation experiments	141
	Introduction to Part III	143
8	Twin experiment on the single-neuron toy model	147
8.1	Twin experiment design	148
8.2	Signal and parameters estimation	149
8.3	Signal prediction results	154
8.4	Discussion and conclusions	157
9	Twin experiments on the BBP single-neuron model	159
9.1	First experiment: fine tuning of EnKF parameters	161
9.1.1	Setting parameters for EnKF	162
9.1.2	Sample EnKF output	163
9.1.3	ANOVA tests results	166
9.2	Second experiment: biased initial condition for parameters	169
9.2.1	Signal estimation results	172
9.2.2	Parameter estimation results	176
9.2.3	Improvement rate of the parameter estimation	180
9.2.4	Validation in the first out-of-sample time window	181
9.3	Discussion and conclusions	182
10	Assimilating experimental data in a realistic BBP model	187
10.1	Blue Brain Project experimental dataset	188
10.1.1	Experimental traces selected for data assimilation	190
10.2	Choosing the parameter search space	192
10.2.1	Likelihood profiles	192
10.2.2	Reducing the parameter search space size	197

10.3	Selecting an effective assimilation method	200
10.4	Results analysis	202
10.4.1	Selection by validation in a forecast-skill sense	206
10.5	Discussion and conclusions	212
List of Figures		217
List of Tables		221
List of Algorithms		223
List of Abbreviations		225
Index		227
Bibliography		229

Introduction

In the last years, recent technological advances and concerted research efforts allowed the international scientific community to address one of the biggest unsolved questions of humanity: how does the human brain work? This was possible thanks to large governmental investments that channelled resources not only towards the theoretical implications of this question (how are memories stored? what are the biological basis of consciousness? what is the role of brain waves?), but also towards the medical and technological applications (e.g. identification of the causes of mental disorders, mind-machine interfaces, brain-inspired computing, etc.). Among other contributors, some of the biggest governmental projects include the European-Union funded Blue Brain Project (BBP) and Human Brain Project (HBP), and the US-based Brain Research through Advancing Innovative Neurotechnologies (BRAIN) initiative. At the end of 2017, representative of these and other Korean, Japanese and Australian brain research projects signed a declaration of intents with the objective of forming a collaborative International Brain Initiative (IBI). The aim is to join their forces in order to “measure, map, image, model, simulate, understand, imitate, diagnose and heal the brain” [1]. In general, the interdisciplinary set of expertise that is required to tackle these extremely challenging and mostly still unsolved questions, which span from biology and neurophysiology to mathematics, statistics and computer science, are the core of the relatively new research area of computational neuroscience.

At the same time, the incredibly large amount of digital data available nowadays has fuelled the research area of machine learning and statistical learning to unprecedented levels of potentiality. However, many of the most popular machine-learning methods only fit deterministic black-box models to specific tasks, without allowing incertitude in either the model or the data, nor providing detailed insights into the phenomenon in exam. In particular, although employed in specific computer-science applications with incredibly results (e.g. face-recognition, data classification, targeted advertising, self-driving cars, etc.), the fashionable deep-learning technique for artificial neural networks (ANNs) often does not add any human-interpretable information to

the world's scientific knowledge. Nonetheless, it is sometimes the case that some mathematical model is deemed suitable to effectively account for some system relevant features, but the experimental data do not match its trajectories.

In such cases, it is possible to resort to Bayesian inference to fit the time course of experimental data in a probabilistic way, while preserving all model characteristics and structure. In fact, the class of Bayesian methods allows one to adjust its prior knowledge on the parameters of a stochastic model in order to make its trajectories fit the experimental data in a probabilistic manner. By doing so, both model errors and measurement errors are taken into account while all original model features are preserved, so that interpretation is still possible. Note that, despite the birth of Bayesian inference dates back to the eighteenth century (Thomas Bayes 1702–1761), its massive application in non-exact form to real-life problems is relatively recent. Nevertheless, data assimilation (DA) (how nowadays Bayesian inference is often referred to) has historically proved to be an indispensable device in the geosciences (e.g. weather forecasting [22, 69, 92, 108, 144], air quality forecasting [26, 130, 222, 223], hydrology [94, 95, 158, 176, 212], oceanography [39, 75, 91, 166, 200], fossil fuels reservoir managing [11, 74, 165, 191, 221, 112] etc.), and has now become a well-established and fully-fledged tool in the hands of practitioners. On the contrary, its use in computational neuroscience is much more recent, and only in the last decades DA has started to show its power to the fast-evolving research area of computational neurobiology. In particular, the last decades witnessed many different works which address the task of parameter estimation in single-neuron models [12, 56, 97, 206, 218], and recent studies suggest it could become a powerful tool for parameter tuning for neuronal assemblies and neural populations as well [161, 40, 137, 188, 141, 213].

The research work I carried out during my PhD course in Pure and Applied Mathematics at Politecnico di Torino places itself at the intersection of these two fast-advancing domains. In fact, under the supervision of Professor Claudio Canuto, it was proposed to me by Professor Wenlian Lu and Professor Jianfeng Feng from Fudan University of Shanghai (China) to jointly work on data assimilation problems in neurobiology modelling. The broad aim of the project was to take advantage of the information contained in experimental data of neurobiological activity at different levels in order to fine-tune mathematical models by applying parameter inference methods. In particular, the focus was on detailed microscopic single-neuron models and on mesoscopic models of neural activity which include the description of the microscopic scale. Now, before overviewing its content, let me highlight that the peculiarity and the motivation behind this thesis work are two-fold, regarding on the

one hand the type of neuronal model we consider, and on the other hand the class of inference methods we apply to estimate both model parameters and state variables dynamics.

First, focusing on the type of models, we select physiologically-detailed conductance-based single-neuron models which explicitly describe the dynamics of ionic currents and voltage-dependent gating variables. This is because neuromodelling has made huge steps forward from the time the original Hodgkin-Huxley (HH) model was proposed in 1952 [89], and nowadays many specific characteristics of a neuron’s electrophysiology (e.g. the presence of depolarization blocks [23], changes in excitability patterns [148], the back-propagation of action-potentials [73, 57], resonance [48], bursting [48, 193] etc.) can be reproduced in the flexible conductance-based framework. If the tuning of such detailed neuronal model gives good parameter accuracy, one can in principle obtain hints about the underlying biological processes (e.g. the presence or absence of some specific ionic current as in [12, Section 5.2.10]) which low-order models such as leaky integrate-and-fire or less detailed HH-type models would not give. In addition, such rich and detailed neuronal models can produce accurate neural networks models when coupled together through synaptic currents [149, 47, 157], and this makes the development of estimation procedures for this type of models worth to be investigated.

However, complex models result in a large number of model parameters (typically, the maximal ionic conductances but also reversal potentials or parameters involved in the activation variables’ dynamics). As a consequence, the search space is large-dimensional and parameter identification is potentially difficult. To address this challenging task various approaches, spanning from linear regression to direct minimization of specific cost functions [96, 195, 85, 151, 132, 80], have been employed over the years. For instance, to fine-tune their multicompartmental neuronal models, the BBP and the HBP currently apply a multi-objective optimization achieved through genetic algorithms [60, 61]. Such procedure aims at fitting the model response to stimuli of different amplitude to the statistics of some electrophysiological features that characterise the neuron’s electrophysiology. To obtain these statistics, the features are automatically extracted and aggregated from various experimental data recorded by different cells (see the Supplementary material of [149]). In our case, we aimed at testing a general inference methodology that easily generalizes to different neurobiological activities and allows a model to be fine-tuned only relying on a given experimental trace, in order to reproduce and predict point-wisely trajectories out-of-sample recordings.

In particular, guided by the experience of previous works which successfully applied Bayesian methods to complex Hodgkin-Huxley-type neuron mod-

els [206, 97, 164, 106], many aspects made us opt for the class of Bayesian DA methods. First, the Bayesian nature of these methods allows one to explicitly include the prior probability distributions for the parameters value and/or for the state variables' initial condition. This is an appealing option which is often not possible with other non-Bayesian parameter estimation methods. In particular, such prior distributions also explicitly include a model for the dynamical and/or data noise. Such characteristic can be particularly suitable to neurobiological application, as it is generally recognized that noise is ubiquitous and plays a major role in the nervous system [70, 186, 194]. In our works, we modelled the dynamical and measurement noise for voltage and activation variables as being Gaussian and white, as typically assumed in single-neurons voltage dynamics [150, 147, 31, 34]. This is a common assumption in DA literature (see for instance [206, 55, 218]) which corresponds to central-limit-theorem like hypothesis on the stochastic process underlying the random noise. Note, however, that pink noise exhibiting $1/f^\alpha$ power spectrum has been proposed to arise from Poisson-like processes (examples include in channel noise, MEG and EEG signals: see [216] and references therein) and that the channel noise would be more realistic if added at the conductance level rather than the activation-variable one [77]. Although we do not consider such cases, the versatile state-space model framework allows one to consider any type of noise deemed realistic.

With this motivation, the first work we completed is a study of the applicability and efficiency of data assimilation methods to a two-dimensional Hodgkin-Huxley-type single neuron [169]. The twin experiment – an assimilation experiment where the dataset is artificially generated by the same model that is subsequently employed to assimilate the data – was chosen to begin with, so that it is granted that the model is able to reproduce the data, and all sort of performance metrics can be assessed. Indeed, in twin experiments the true parameter values that generated the data are known, which is not normally the case in the truly experimental case. In this preliminary work, we tested and compare the estimation results obtained by three different on-line filters – namely the bootstrap filter (BF), its optimal version, i.e. the optimal sequential importance resampling (OPT-SIRS), and the ensemble Kalman filter (EnKF). In particular, we checked not only the accuracy in estimating ten model parameters (which include maximal conductances, reversal potentials and parameters involved in the ionic currents dynamics) but also the ability of the resulting model to predict the neuron response to new stimuli.

Once we observed that the methods under investigation could indeed address the task of parameter tuning and signal prediction in a well-known mathematical model of single neurons, we moved to a more realistic model. In par-

ticular, we considered the morphologically- and electrophysiologically-detailed models proposed by the BBP in their *in silico* reconstruction and simulation of a juvenile Wistar rat neocortical microcircuit [149]. Of all the models they proposed, we focused on the somatic compartment of the model for the pyramidal cells of the second and third cortical layer. On such model, we first ran two preliminary twin experiments applying EnKF.

In the first one, we look for the EnKF setting parameters that produce the best signal predictions and the best estimation of the seven parameters (five maximal conductances and two parameters involved in the internal calcium dynamics). In the second twin experiment we make a step towards assimilating real data by investigating whether or not a biased initial parameter guess still allows good estimations and predictions. Indeed, when dealing with experimental data, not knowing the “true” modelling parameters means that it is likely to initialise the algorithm away from a good set of parameters.

In these two twin experiments, we also propose and use an EnKF extension (which technically only applies to unbounded state variables and parameters) to the bounded and half-bounded case. This is possible by resorting to change of variables between the physical bounded set and the real line and applying the EnKF to the unbounded transformed variables. Of course, this distorts the noise distribution, but we show that the distortion effect in the bounded case (i.e. in case of logit-normal noise) is only significant at the edges of the bounded set, at least when the variance is small.

At last, we considered some truly experimental single-neuron recordings and proceeded in assimilating them in the above-mentioned BBP realistic neuron model. Disappointingly, we failed in successfully applying the EnKF to the data. Nonetheless, by analysing the likelihood profiles we were able to shrink the parameter search space to the region where the model exhibits meaningful neuron-like responses to the experimental input. Relying on such reduced search spaces, we were finally able to apply swarm optimization techniques which allowed us to find some parameter estimates that make the model fit the experimental spike trains, and reproduce out-of-sample experimental data to some extent. However, we do not consider the results we found completely satisfactory, and we then propose some possible explanations in the very last section.

Finally, we here report that we also started the data assimilation project on a mesoscopic neurobiological model which include a microscopic detail. Namely, we began investigating the possibility of using measurements of the collective activity of a neural network – either invasive, such as local field potential (LFP), or non-invasive such as electroencephalography (EEG), magnetic resonance imaging (MRI), or functional MRI (fMRI) – in order to estimate the

microscopic parameters which are relative to the network units. In particular, the model of neural tissue we consider is formed by leaky integrate-and-fire (LIF) neuron models which interact through chemical synapses. Because of the large number of computational units in biological neural networks, we assume that all parameters in a given region follow the same probability distribution, and what we aim at estimating are the hyperparameters of such distribution. Although this project is still at a preliminary phase, we envision that such an inference framework could be employed to deduce the microscopic features of *in-vivo* neural tissues by only using non-invasive recordings of the brain activity, which would represent a powerful tool in clinical studies.

This thesis is composed of three main parts: in Part I we introduce in some detail the data assimilation methods which were employed in our numerical experiments; Part II illustrates all the mathematical models of neuroactivity that have been considered in the whole project; finally, in Part III we present the results of all inference experiments we carried out. For a detailed presentation and thorough breakdown of each part, we refer to the relative introductions.

Part I

Data assimilation methods

Introduction to Part I

The first part of this thesis is devoted to the mathematical description of the algorithms which are employed in Part III to perform parameter fine-tuning in the single-neuron models we introduce in Chapter 5. The methods we present go under the non-specific name of data assimilation algorithms (DA), but a number of other names have been used in different scientific communities. For instance, in the statisticians community hidden Markov models, Monte Carlo methods, and self-organizing state-space models are terms widely used to denote either the data assimilation model framework, or particular families of methods. On the other hand, the control engineers and mathematicians community often refers to data assimilation problems with the term inverse problems. This is to stress that in inverse problems the aim is to identify the parameters of a given model from its output, whereas in direct problems the output is generated by a model once its initial conditions and its parameters are set. However, independently on how one refers to such methods, data assimilation designates a family of algorithms which has been employed in the last decades to carry out the task of hidden-variables estimation and parameters assessment in a number of different domains. Examples include meteorology [109], geochemistry [63], systems biology [138], econometrics [140], petroleum industry applications [165], oceanography [92], and financial mathematics [154].

As far as we are concerned, in this part we present DA by overviewing different class of methodologies and specific instances of such algorithms. There are two key concepts underpinnings such methods: the Bayesian framework and the dynamical structure underpinning the state-space methodology. First, we can synthetically describe the Bayesian framework of DA methods as assuming that all unknown quantities which are to be estimated are subsumed in some signal random variable Z which follows a prior distribution $p(z)$. This prior distribution represents the knowledge on has *a priori* on the system in exam. Then, it is supposed that the system can be observed through another random variable Y , which is typically lower dimensional and is linked to the signal random variable through the likelihood distribution ($p(y|z)$), so that the

Bayes' theorem ($p(z|y) \propto p(z)p(y|z)$) can be applied to compute or approximate the posterior distribution with some specific DA method. On the other hand, the dynamical nature of data assimilation means that the signal variable is a stochastic process, either discrete or continuous in time: $Z = (Z_j; j \in \mathbb{N}_0)$ or $Z = (Z_t; t \in \mathbb{R}^+)$, respectively. As a consequence, the aim of Bayesian DA is to point-wisely estimate the hidden states underlying the realisation of the corresponding data trajectory. In our case, we only address Bayesian data assimilation in discrete-time (i.e., the assimilated data consist in a observed time series ($Y_j = y_j; j \in \mathbb{N}_0$)) applied to dynamical systems describing neurobiology processes. Nevertheless, the data assimilation methodology applies to more general stochastic processes.

Overall, Part I is organised as follows. In Chapter 1, we present the state-space model framework used to represent the dynamic phenomenon from a probabilistic point of view. Then, we move to a survey of the most popular examples of data assimilation methods, going into the details of some notable algorithms. Following [134], such overview is subdivided into the two main families of algorithms: the family of off-line smoothing methods and the family of filtering algorithms. The former is addressed in Chapter 2, which opens up by formally defining the smoothing distribution and illustrating such definition in the nonlinear Gaussian case. To be more specific, we describe some instances of the Markov chain Monte Carlo methodology whose objective it to approximate the smoothing distribution and is effectively applicable to non-Gaussian models (Section 2.1) as well as variational methods which aim at directly maximizing the posterior distribution to identify its modes (Section 2.2). On the other hand, in Chapter 3 we present the on-line approach characterizing filtering methods, which are the algorithms we test the most in the numerical experiments described in PartIII. We firstly focus on the well-performing family of Gaussian-like filters (Section 3.1) and then on the more theoretically justified particle filters which can be considered the sequential version of Monte Carlo Markov chain methods (Section 3.2).

Finally, Chapter 4 addresses a series of issues which may arise when applying data assimilation methods to real-life problems. The first issue we discuss is how to include static parameter and hyperparameter estimation into the state-space model framework introduced in the first chapter (Section 4.1). In particular, Section 4.1.1 lists different approaches which can be used for parameter estimation (e.g., augmenting the state-space, exploiting expectation-maximization, and many more), and Section 4.1.2 shows how to extend the augmented state-space approach whenever it suffice to estimate the hyperparameters which characterise the probability density of many identically-distributed parameters. Then, in Section 4.2 we show how to build

the discrete-time state-space model step by step in case the proposed model is given in continuous-time form (i.e. in form of ordinary differential equation). Afterwards, in Section 4.3 we propose a way to adapt the Gaussian state-space model formalism to cover the case of a “prior” model with hard constraints and also investigate what biases such adaptation introduces. In conclusion, in Section 4.4 we report a brief overview based on [134, Section 2.7] regarding the different approaches one can adopt to evaluate the quality of data assimilation methods, and then propose a performance score which can be used whenever one adopts a forecast-skill approach in a twin-experiment framework (Section 4.4.1).

Chapter 1

Generalities on state-space models

In the current chapter, we simply give the exact definition of state-space model, and illustrate such definition with a couple of examples which are paradigmatic and frequently recur in the rest of this work.

The first base concept to introduce in order to define **data assimilation** methods is the notion of state-space model. The state-space model formalism has been extensively used in the DA literature since thanks to its generality it is possible to cover the description of many different dynamical phenomena. Indeed, the fundamental idea which SSMs formalise is that whenever we can observe some data concerning a given system (e.g. atmospheric, biological, economic, etc.), it is often the case that what we observe does not provide a complete characterization of the physical quantities describing the system. In such circumstances, considering the data as a simple projection of the more complex dynamics on a easily observable space can give a significant insight on the complete phenomenon. Let us now define such intuitive idea in a precise mathematical manner.

1.1 The state-space model framework

A discrete-time **state-space model** (SSM) is a couple of two stochastic processes: a hidden signal process $(Z_j ; j \in \mathbb{N}_0)$ and a measurable observation process $(Y_j ; j \in \mathbb{N}_0)$. Notice that we denote the strictly positive integers $\mathbb{N} = \{1, 2, 3, \dots\}$, whereas we write $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$ to indicate the non-negative integers $\{0, 1, 2, \dots\}$.

In particular, the **signal process** is a d_z -dimensional non-stationary Markov

chain $(Z_j; j \in \mathbb{N}_0)$ with state space $\mathcal{Z} \subset \mathbb{R}^{d_z}$. We denote $f_0(z_0)$ its initial density and, for $j > 0$, $f_{j-1}(z_j|z_{j-1})$ the non-homogeneous (i.e. time-dependent) transition density. This is to say that the signal process is given by

$$\begin{cases} Z_0 \sim f_0, \\ Z_j \mid (Z_{0:j-1} = z_{0:j-1}) \sim f_{j-1}(\cdot \mid z_{j-1}), \quad j > 0. \end{cases} \quad (1.1)$$

Remark 1.1 (Essential summary on probability notation).

Throughout this text, for any given random variable X , the notation $X \sim \mathcal{L}$ is used to indicate that the random variable X is distributed according to the probability distribution \mathcal{L} . On the other hand, in the following we sometimes write $X = Y + Z$ for three arbitrary random variables X, Y , and Z . In such cases the equal sign means that, given the realizations of the random variables in the right-hand side, one can uniquely determine the realization of the random left-hand side.

Also, we adopt the convention that capital letters denote random variables, whereas lowercase letters indicate their realisations (e.g., $X = x$ means that the specific value x is the realisation of the random variable X) or else, purely deterministic variables. The only exception to such rule are random variables denoted by Greek letters, such as the noise random variables ϵ and ε we introduce in Example 1.2 and Example 1.3 or the parameter random vector θ we first mention in Chapter 4. Note that it is a standard convention that random variables denoted by Greek letters are not capitalised, and the same symbol is often used to indicate their realizations if explicitly stated. Nevertheless, if the context is not clear and ambiguity is possible we introduce a different symbol to indicate the realization of Greek-letter-denoted r.v.'s, the main instance being $\theta = \check{\theta}$ in Section 4.1 which means, in such specific context, that the value $\check{\theta} \in \Theta$ is the realization of the random variable θ .

In addition, remark that we use the same symbol for probability distributions and the respective density functions, so that $\mathcal{L}(x)$ (with the function argument explicitly stated) denotes the p.d.f. of a random variable X which is distributed according to the probability law \mathcal{L} (where no argument is present as the symbol denotes a distribution rather than a function). An example of such non-standard notation is the following. Suppose X is a (scalar) Gaussian random variable of mean μ and variance σ^2 . Then $X \sim \mathcal{L}$, where the probability distribution is $\mathcal{L} = \mathcal{N}(\mu, \sigma^2)$, and the respective density function is denoted

$$\mathcal{L}(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right).$$

Finally, $X_{i:j}$ is a compact notation for $(X_i, X_{i+1}, \dots, X_{j-1}, X_j)$, for any integer time indices i and j and for any stochastic or deterministic variable. ♠

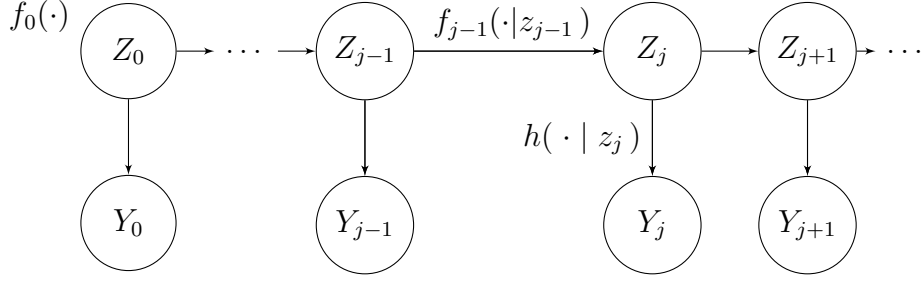


Figure 1.1: Diagram of the general state-space model defined by (1.1)-(1.2).

On the other hand, the **observation process** is a stochastic process $(Y_j ; j \in \mathbb{N}_0)$ with state space $\mathcal{Y} \subset \mathbb{R}^{d_y}$, where typically $d_y < d_z$. The conditional density of the observation process is defined by

$$Y_j \mid (Z_{0:j} = z_{0:j}, Y_{0:j-1} = y_{0:j-1}) \sim h(\cdot \mid z_j), \quad j \geq 0, \quad (1.2)$$

and a given realization of the observation process $Y_{0:j} = y_{0:j}$ is called **data**.

Note that the second equation of (1.1), we implicitly make use of the forgetting property of Markovian processes. In fact, it is implied that the random variable $Z_j \mid (Z_{0:j-1} = z_{0:j-1})$ only depends on the chain state at the previous time step $Z_{j-1} = z_{j-1}$, and it does not depend on the preceding states $Z_{0:j-2} = z_{0:j-2}$.

In addition, we can rewrite (1.1) in the following alternative form

$$\begin{cases} p(z_0) = f_0(z_0) \\ p(z_j \mid z_{0:j-1}) = f_{j-1}(z_j \mid z_{j-1}), \end{cases} \quad (1.3)$$

where the Markov property consists in stating that $p(z_j \mid z_{0:j-1}) = p(z_j \mid z_{j-1})$. Note that we just introduced a further notation which is used throughout this text. Namely, whenever we write $p(x)$ we intend the probability density function of some random variable X (or, occasionally, its distribution), and specifying $p(x) = f(x)$ gives the exact functional form of the density function. For conditional random variables, writing $p(x \mid y) = f(x \mid y)$ means that the two-arguments function $f(x \mid y)$ is the density of the r.v. $X \mid (Y = y)$.

Using the same notation, (1.2) can be rewritten as

$$p(y_j \mid z_{0:j}, y_{0:j-1}) = h(y_j \mid z_j), \quad (1.4)$$

where $h(y_j \mid z_j) = p(y_j \mid z_j)$.

In Figure 1.1, we give a graphical representation of a general state-space model. Such representation is given in the form of a **directed acyclic graph**

(DAG). For a formal definition and an introduction to DAGs see, for instance, [102, Chapter 2]. The first row of nodes in the figure represents the signal process $(Z_j; j \in \mathbb{N}_0)$, which we cannot observe directly. On the other hand, the sequence of nodes at the bottom represents the observable data process $(Y_j; j \in \mathbb{N}_0)$. The arrows pointing downward are labelled with the observation conditional density $h(\cdot | z_j)$, which links the signal variable Z_j to the corresponding observation Y_j . Finally, the horizontal arrows represent the Markovian transitions of the signal process $(Z_j; j \in \mathbb{N}_0)$.

In general, the aim of data assimilation is to use the information contained in a given realization of the observation process $(Y_j = y_j; j \in \mathbb{N}_0)$ – which is to say, to make use of the data – in order to infer information on the latent signal process $(Z_j; j \in \mathbb{N}_0)$, which is not directly observable. Such layered structure is what motivates the alternative name **hidden Markov model** for state-space models.

Let us now consider some notable example of state-space models to which we systematically refer to throughout this text.

Example 1.2 (Gaussian linear model).

A rather simple example of state-space model is the Gaussian linear model. In such model, the signal space is the whole d_z -dimensional hyperspace $\mathcal{Z} = \mathbb{R}^{d_z}$, and the signal model is given by

$$\begin{cases} Z_0 \sim \mathcal{N}(\mu_0, C_0) \\ Z_j \mid (Z_{0:j-1} = z_{0:j-1}) \sim \mathcal{N}(Mz_{j-1}, \Sigma_z), \end{cases}$$

where M is a $d_z \times d_z$ matrix (i.e., a linear operator).

The observation model is defined by

$$Y_j \mid (Z_j = z_j) \sim \mathcal{N}(Hz_j, \Gamma),$$

where H is a $d_y \times d_z$ matrix mapping the signal space \mathcal{Z} into the observation space $\mathcal{Y} = \mathbb{R}^{d_y}$.

We can reformulate such state-space model in the following way. Given a Gaussian initial condition $Z_0 \sim \mathcal{N}(\mu_0, C_0)$, we define the stochastic dynamical system

$$Z_j = MZ_{j-1} + \epsilon_j, \quad j > 0,$$


iteratively in time. The **dynamical noise process** $(\epsilon_j)_{j>0}$ is an independent identically-distributed (i.i.d.) sequence with the first element distributed as

$\epsilon_1 \sim \mathcal{N}(0, \Sigma_z)$ which is independent on the initial condition Z_0 . With such alternative definition, conditioning on $Z_{j-1} = z_{j-1}$, the Markovian dynamics is given by the perturbation of the **deterministic discrete-time map** $g_{j-1}(z_{j-1}) = Mz_{j-1}$ plus the additive random noise ϵ_j .

In a similar fashion, the conditional density for an observation Y_j can be interpreted as the sum of the linear **deterministic observation operator** $H : \mathcal{Z} \rightarrow \mathcal{Y}$ and a Gaussian measurement noise ε_j , i.e.

$$Y_j = HZ_j + \varepsilon_j.$$

The **measurement noise process** $(\varepsilon_j)_{j \geq 0}$ is an i.i.d. random sequence with $\varepsilon_0 \sim \mathcal{N}(0, \Gamma)$, which is independent on both the initial condition Z_0 and the stochastic process $(\epsilon_j)_{j > 0}$.

In case of a Gaussian linear state-space model, both signal and observation models are linear (i.e. $g_{j-1} : \mathcal{Z} \rightarrow \mathcal{Z}$ and $H : \mathcal{Z} \rightarrow \mathcal{Y}$ are linear operators). In addition, the initial condition Z_0 , and the noise processes $(\epsilon_j)_{j > 0}$ and $(\varepsilon_j)_{j \geq 0}$ are all Gaussian random variables. Because of its analytical tractability, this is a remarkable example which returns in the description of some DA methods in the forthcoming chapters. 

The linear Gaussian model discussed above can be rather easily generalised to its nonlinear counterpart, which is much more useful in practical applications.

Example 1.3 (Gaussian nonlinear model).

The Gaussian nonlinear state-space model has the same signal space $\mathcal{Z} = \mathbb{R}^{d_z}$ as the previous linear SSM, but the deterministic discrete time map $g_{j-1}(z_{j-1})$ and the deterministic observation operator $H(z_j)$ are now allowed to be nonlinear functions.

A recursive definition for the signal dynamics is

$$\begin{cases} Z_0 \sim \mathcal{N}(\mu_0, C_0) \\ Z_j \mid (Z_{j-1} = z_{j-1}) \sim \mathcal{N}(g_{j-1}(z_{j-1}), \Sigma_z), \end{cases} \quad (1.5)$$

where – as before – the dynamical noise process $(\epsilon_j)_{j > 0}$ is an i.i.d. sequence with $\epsilon_1 \sim \mathcal{N}(0, \Sigma_z)$ independent on the initial condition Z_0 .

Also, the observation model is

$$Y_j = H(Z_j) + \varepsilon_j, \quad (1.6)$$

where the measurement noise process $(\varepsilon_j)_{j \geq 0}$ is an i.i.d. random sequence with $\varepsilon_1 \sim \mathcal{N}(0, \Gamma)$, which is independent on both the initial condition Z_0 and the stochastic process $(\epsilon_j)_{j > 0}$.

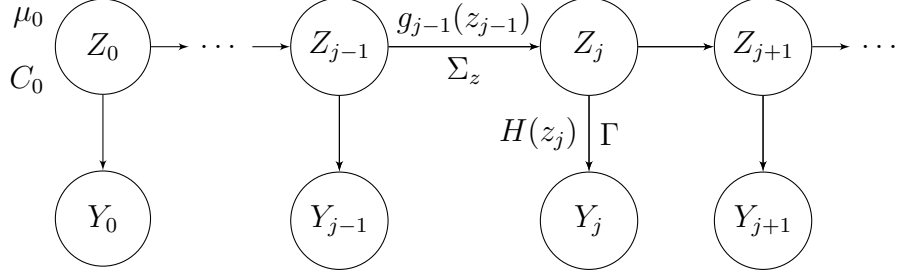


Figure 1.2: DAG of the Gaussian nonlinear state-space model described in Example 1.3

To be completely explicit, the analytical expression for the initial condition probability density function (p.d.f.) is given by

$$f_0(z_0) = \frac{1}{\left((2\pi)^{d_z} |C_0|\right)^{1/2}} \exp\left(-\frac{1}{2} |z_0 - \mu_0|_{C_0}^2\right),$$

and, conditional on $Z_{0:j-1} = z_{0:j-1}$, the signal model transition density has the form of a Gaussian density with mean $g_{j-1}(z_{j-1})$ and covariance matrix Σ_z , i.e.

$$f_{j-1}(z_j | z_{j-1}) = \frac{1}{\left((2\pi)^{d_z} |\Sigma_z|\right)^{1/2}} \exp\left(-\frac{1}{2} |z_j - g_{j-1}(z_{j-1})|_{\Sigma_z}^2\right).$$

Finally, conditional on $Z_{0:j} = z_{0:j}$, the observation model density is Gaussian with mean $H(z_j)$ and covariance matrix Γ , namely

$$h(y_j | z_j) = \frac{1}{\left((2\pi)^{d_y} |\Gamma|\right)^{1/2}} \exp\left(-\frac{1}{2} |y_j - H(z_j)|_{\Gamma}^2\right).$$

Note that in general, given a $\ell \times \ell$ non-singular matrix A , throughout this text we adopt the notation $|\cdot|_A$ to indicate the A -induced norm on \mathbb{R}^ℓ . Such norm is defined by

$$|z|_A = \sqrt{z^T A^{-1} z},$$

for all $z \in \mathbb{R}^\ell$. ♣

Figure 1.2 illustrates the state-space model described in the last Gaussian nonlinear case. As in Figure 1.1, the top row of nodes represents the hidden signal process and the bottom sequence the observation process. Since this is a Gaussian state-space model, only the mean and the covariance matrix of the

probability distributions are marked. In fact, the symbols at the top left corner of the figure mean that the random initial condition is distributed according to $Z_0 \sim \mathcal{N}(\mu_0, C_0)$. The conditional transition distribution is $Z_j | (Z_{j-1} = z_{j-1}) \sim \mathcal{N}(g_{j-1}(z_{j-1}), \Sigma_z)$, and the conditional observation distribution is $Y_j | (Z_j = z_j) \sim \mathcal{N}(H(z_j), \Gamma)$. This concludes preliminary introduction to state-space models.

In the next two chapters, we present in detail a series of data assimilation algorithms which in Part III are applied to different models of neurobiological activity. Note that the family of DA methods can be subdivided into two main classes: the class of off-line smoothing algorithms, which we address in Chapter 2, and the class of on-line filtering algorithms, discussed in Chapter 3. We now proceed presenting the main features of each of these two groups, and we introduce some notable examples for each class of algorithms. Note that in both cases, we consider a fixed time horizon $J \in \mathbb{N}$ and the resulting finite time window $\{0, \dots, J\}$. We hereafter refer to such discrete time interval as the **data assimilation time window**.

Chapter 2

Off-line smoothing methods

In this chapter, first we introduce the smoothing problem, and then we present its negative-log formulation which applies to Gaussian state-space models. Then, we present in detail two different methodologies which belong to the class of smoothing methods: the Markov chain Monte Carlo (Section 2.1), and the class of variational algorithms (Section 2.2). The latter is exemplified in Section 2.2.2 by presenting the freely-available implementation of the four-dimensional variational algorithm.

Considering a Bayesian framework, we present the **smoothing problem** as follows. First, let us introduce the **prior density** $p(z_{0:J})$. Such quantity represents the prior knowledge that one can deduce from the signal model (1.1) about the signal variable $Z_{0:J}$. Taking advantage of the signal model definition given in (1.3), the prior $p(z_{0:J})$ can be explicitly computed as

$$\begin{aligned} p(z_{0:J}) &= p(z_J|z_{0:J-1})p(z_{0:J-1}) \\ &= p(z_J|z_{J-1})p(z_{0:J-1}) \\ &= \dots \\ &= p(z_0) \prod_{j=1}^J p(z_j|z_{j-1}) \\ &= f_0(z_0) \prod_{j=1}^J f_{j-1}(z_j|z_{j-1}), \end{aligned} \tag{2.1}$$

where we solely applied the Markov property while resorting to recursion.

It is somewhat customary to write the prior as

$$p(z_{0:J}) = \exp\left(-\mathcal{P}(z_{0:J})\right),$$

where the **negative-log prior** $\mathcal{P} : \mathcal{Z}^J \rightarrow \mathbb{R}$ is tautologically defined by

$$\mathcal{P}(z_{0:J}) := -\log \left(p(z_{0:J}) \right). \quad (2.2)$$

Hence, taking advantage of the homomorphy property¹ of the logarithm function, (2.1) yields

$$\mathcal{P}(z_{0:J}) = -\log \left(f_0(z_0) \right) - \sum_{j=1}^J \log \left(f_{j-1}(z_j | z_{j-1}) \right).$$

Note that the above expressions are actually meaningful only when the prior distribution is strictly positive on the whole signal state space, i.e. $p(z_{0:J}) > 0$ for all $z_{0:J} \in \mathcal{Z}^{J+1}$.

Now, suppose we are given a set of data in the data assimilation time window $\{0, \dots, J\}$, namely $(Y_j = y_j ; j = 0, \dots, J)$, a given realization of the observation process. Then, the conditional density $p(y_{0:J} | z_{0:J})$ is called the **likelihood function** (or simply the likelihood) and, using the observation model definition (1.4), it can be computed as

$$\begin{aligned} p(y_{0:J} | z_{0:J}) &= p(y_J | z_{0:J}, y_{0:J-1}) p(y_{0:J-1} | z_{0:J}) \\ &= p(y_J | z_J) p(y_{0:J-1} | z_{0:J}) \\ &= \dots \\ &= \prod_{j=0}^J p(y_j | z_j) \\ &= \prod_{j=0}^J h(y_j | z_j). \end{aligned} \quad (2.3)$$

As for the prior, introducing the **negative-log likelihood**

$$\begin{aligned} \mathcal{L}(z_{0:J}; y_{0:J}) &:= -\log \left(p(y_{0:J} | z_{0:J}) \right) \\ &= -\sum_{j=0}^J \log \left(h(y_j | z_j) \right), \end{aligned} \quad (2.4)$$

the likelihood function is given by

$$p(y_{0:J} | z_{0:J}) = \exp \left(-\mathcal{L}(z_{0:J}; y_{0:J}) \right),$$

¹If $a, b > 0$, then $\log(ab) = \log(a) + \log(b)$.

where $p(y_{0:J}|z_{0:J}) > 0$, for all $z_{0:J} \in \mathcal{Z}^{J+1}$ and $y_{0:J} \in \mathcal{Y}^{J+1}$.

Taking advantage of the likelihood function, the posterior **smoothing density** $p(z_{0:J}|y_{0:J})$ can be computed applying the Bayes' formula

$$p(z_{0:J}|y_{0:J}) = \frac{p(y_{0:J}|z_{0:J})p(z_{0:J})}{p(y_{0:J})}. \quad (2.5)$$

Remark 2.1. *Thinking of the smoothing density as a function of the signal $z_{0:J} \mapsto p(z_{0:J}|y_{0:J})$, the denominator $p(y_{0:J})$ of the right hand side (r.h.s.) of (2.5) can be thought as a normalizing constant which can be easily computed integrating $z_{0:J}$ out of numerator. In fact, since the smoothing density $p(z_{0:J}|y_{0:J})$ has unitary mass, $p(y_{0:J}) = \int_{\mathcal{Z}^{J+1}} p(y_{0:J}|z_{0:J})p(z_{0:J}) dz_{0:J}$. Because of such observation, Bayes' formula is often written in the alternative form*

$$p(z_{0:J}|y_{0:J}) \propto p(y_{0:J}|z_{0:J})p(z_{0:J}), \quad (2.6)$$

which reads: “the posterior distribution is proportional to the prior distribution times the likelihood”.

However, the computation of the normalizing constant in (2.6) can be computationally expensive in practice, especially when the state space is high-dimensional. This is the case, for instance, whenever the signal is high dimensional (large d_z) or for long temporal windows (large J). In such situations, it can be inefficient to compute or sample directly the exact smoothing distribution. Precisely this reason motivates the need of smoothing algorithms capable of approximating the smoothing distribution. ♠

In conclusion, the posterior density for the smoothing problem set by (1.1) and (1.2) is given by

$$p(z_{0:J}|y_{0:J}) \propto \overbrace{f_0(z_0) \prod_{j=1}^J f_{j-1}(z_j|z_{j-1})}^{\text{prior}} \overbrace{\prod_{j=0}^J h(y_j|z_j)}^{\text{likelihood}}. \quad (2.7)$$

In addition, defining the **negative-log smoothing density** as

$$\begin{aligned} \mathcal{S}(z_{0:J}; y_{0:J}) &:= -\log \left(p(z_{0:J}; y_{0:J}) \right) \\ &= \mathcal{P}(z_{0:J}) + \mathcal{L}(z_{0:J}; y_{0:J}) + \text{const.} \end{aligned} \quad (2.8)$$

the smoothing density can be written as

$$p(z_{0:J}|y_{0:J}) = \exp \left(-\mathcal{S}(z_{0:J}; y_{0:J}) \right). \quad (2.9)$$

Since – as mentioned in Remark 2.1 – normalizing constants are often not relevant in our discussion, in the rest of this text we will omit the additive constant appearing in (2.8). Furthermore, it will be always implied that the definitions for the neg-log prior (2.2) and the neg-log likelihood (2.4) hold up to additive constants.

Such expressions for the prior, the likelihood, and smoothing distribution in negative-log form are mostly useful in case of Gaussian probability densities, as we show in the following example.

Example 2.2 (Smoothing distribution in the Gaussian case).

In the Gaussian nonlinear state-space model presented in Example 1.3, the prior density is given by

$$\begin{aligned}
p(z_{0:J}) &= f_0(z_0) \prod_{j=1}^J f_{j-1}(z_j|z_{j-1}) \\
&\propto \exp\left(-\frac{1}{2}|z_0 - \mu_0|_{C_0}^2\right) \prod_{j=1}^J \exp\left(-\frac{1}{2}|z_j - g_{j-1}(z_{j-1})|_{\Sigma_z}^2\right) \\
&\propto \exp\left(-\frac{1}{2}|z_0 - \mu_0|_{C_0}^2\right) \exp\left(-\frac{1}{2}\sum_{j=1}^J |z_j - g_{j-1}(z_{j-1})|_{\Sigma_z}^2\right) \\
&\propto \exp\left(-\frac{1}{2}|z_0 - \mu_0|_{C_0}^2 - \frac{1}{2}\sum_{j=1}^J |z_j - g_{j-1}(z_{j-1})|_{\Sigma_z}^2\right).
\end{aligned}$$

Thus, the negative-log prior defined in (2.2) in the Gaussian case is

$$\mathcal{P}(z_{0:J}) = \frac{1}{2}|z_0 - \mu_0|_{C_0}^2 + \frac{1}{2}\sum_{j=1}^J |z_j - g_{j-1}(z_{j-1})|_{\Sigma_z}^2. \quad (2.10)$$

On the other hand, the likelihood function writes

$$\begin{aligned}
p(y_{0:J}|z_{0:J}) &= \prod_{j=0}^J h(y_j|z_j) \\
&\propto \prod_{j=0}^J \exp\left(-\frac{1}{2}|y_j - H(z_j)|_{\Gamma}^2\right) \\
&\propto \exp\left(-\frac{1}{2}\sum_{j=0}^J |y_j - H(z_j)|_{\Gamma}^2\right),
\end{aligned}$$

or, equivalently, it is given by $p(y_{0:J}|z_{0:J}) = \exp\left(-\mathcal{L}(z_{0:J}; y_{0:J})\right)$, where

$$\mathcal{L}(z_{0:J}; y_{0:J}) = \frac{1}{2} \sum_{j=0}^J |y_j - H(z_j)|_{\Gamma}^2. \quad (2.11)$$

In conclusion, the smoothing density is given by (2.9), where

$$\mathcal{S}(z_{0:J}; y_{0:J}) = \frac{1}{2} |z_0 - \mu_0|_{C_0}^2 + \frac{1}{2} \sum_{j=1}^J |z_j - g_{j-1}(z_{j-1})|_{\Sigma_z}^2 + \frac{1}{2} \sum_{j=0}^J |y_j - H(z_j)|_{\Gamma}^2 \quad (2.12)$$

♣

Smoothing algorithms are **off-line** inference methods, which is to say that we need to condition over the whole dataset in order to infer information on the signal Z_j at any time step j . In fact, once the smoothing density $p(z_{0:J}|y_{0:J})$ is known, its marginal distribution can also be computed by integration with respect to all other variables

$$p(z_j|y_{0:J}) = \int_{\mathcal{Z}^J} p(z_{0:J}|y_{0:J}) \, dz_{0:j-1} \, dz_{j+1:J}. \quad (2.13)$$

The above expression contains the whole information on Z_j which can be drawn from the dataset $Y_{0:J} = y_{0:J}$. However, it should be emphasised that in (2.13), in order to obtain information on the signal at time j , we also condition on “future” data $y_{j+1}, y_{j+2}, \dots, y_J$.

Being an off-line method is not normally a flaw. On the contrary, since even future point of the trajectory are used, using a smoothing algorithm is usually very effective in most cases. However, conditioning on the whole dataset at each time step is often computationally expensive and results in large CPU time even for relatively small problems, as we show in [169].

In addition, the off-line approach may be problematic whenever the estimation procedure should be performed “on the fly”, as data arrive. This is the case, for instance, in weather forecast applications: at the present day, when a new atmospheric measurement is recorded, one wishes to run a new inference step while preserving the estimation made the day before. In such situations, whenever a new datum is available, off-line methods need to run the algorithm with a new augmented dataset from scratch. On the contrary, it is possible to tackle the estimation of the posterior distribution $p(z_{0:J}|y_{0:J})$ one time-step at the time, by taking advantage of **on-line** algorithms constructed in a sequential manner. Such algorithms are commonly called filtering methods and they are discussed in detail in the next chapter.

But first, we continue our presentation of smoothing methods by introducing some classical algorithms. In particular, we focus on two families of methods: the Markov chain Monte Carlo methodology, introduced in the following section, and the variational methodology, discussed in the subsequent Section 2.2. For each one of these two approaches, we illustrate some specific examples and give the complete algorithm in the case of Gaussian nonlinear model.

2.1 The Markov chain Monte Carlo methodology

We now present the Markov chain Monte Carlo (MCMC) methodology, which can be applied to approximate any given probability distribution π on a given state space $\mathcal{X} \subset \mathbb{R}^\ell$. As a consequence, in the current section we drop the terminology established for the smoothing problem and only refer to a general probability density π , so to stress the generality of such methodology. Its application to the smoothing problem is easily recovered by taking π to be the smoothing density $p(z_{0:J}|y_{0:J})$.

The core idea of **Markov chain Monte Carlo** methods is to consider a discrete-time Markov chain $(X^{(n)})_{n \geq 0}$, with state-space $\mathcal{X} \subset \mathbb{R}^\ell$ and **transition kernel**² $K(x, d\hat{x})$, which admits the target distribution π as an **invariant distribution**³. Such a Markov chain is particularly useful, as it can be proved that under standard ergodic assumptions on $(X^{(n)})_{n \geq 0}$, the chain distribution converges to the (unique) invariant distribution, independently of the initial configuration.

In the general setting of a discrete-time Markov chain with uncountable state space, the typical **standard ergodic assumptions** are related to the following properties:

- i*) **irreducibility** (from any state there is a positive probability to reach any other state);
- ii*) being **recurrent** (in average, the chain visits any possible state an infinite number of times);
- iii*) being **aperiodic** (there is no cyclic structure in the chain trajectories).

²See [179, Definition 4.2.1] for a formal definition of transition kernel for Markov chains.

³See [179, Definition 4.5.1].

We do not provide a precise mathematical definition of such three properties, but the reader may refer to [179, Definition 4.3.1], [179, Definition 4.4.5, Definition 4.4.8], and [179, definitions at page 150], respectively. In addition, refer to [179, Theorem 4.6.5] for a rigorous list of the hypothesis guaranteeing convergence to the invariant distribution in case of countable state space, and to [156, Chapter 13] for a rigorous proof in case of general state space.

Note that the name ergodic for (i)-(iii) is due to the fact that, under such hypotheses, not only the Markov chain is asymptotically distributed as the invariant distribution, but it also satisfies the following ergodic property. Let $(X^{(n)})_{n \geq 0}$ be a discrete-time \mathcal{X} -valued Markov chain with π as invariant distribution. Then, $(X^{(n)})_{n \geq 0}$ is said to be **ergodic** if for all π -integrable functions⁴ $\varphi : \mathcal{X} \rightarrow \mathbb{R}$, the limit

$$\frac{1}{N} \sum_{n=0}^{N-1} \varphi(X^{(n)}) \xrightarrow{N \rightarrow \infty} \int_{\mathcal{X}} \varphi(x) \pi(dx), \quad (2.14)$$

holds for π -almost every (a.e.) initial condition $X^{(0)} = x^{(0)}$.

Such property can be summarised saying that, for ergodic Markov chains time averages – left hand side (l.h.s.) of (2.14) – and space averages – the r.h.s. – are exchangeable for large N .

Remark 2.3. *The **ergodic theorem** is the result stating the sufficient conditions (usually, irreducibility and being recurrent) which guarantee that a Markov chain is ergodic. First, it has been proved by in the context of differential dynamical systems on closed manifolds by G. Birkhoff [24]. However, in modern literature it is usually justified invoking the Law of large numbers, with the Central limit theorem providing the convergence rate.*

We do not explicitly state such result nor we specify its hypotheses as a complete dissertation on the mathematical foundations of MCMC methods goes beyond the scope of this thesis. In particular, note that we did not even indicate the type of convergence in (2.14). For further details on the ergodic theorem and details on its proof see, for instance, [179, Theorem 4.7.4, Proposition 4.7.5]. ♠

Along with the result of convergence to the invariant distribution, the ergodic property is the fundamental theoretical tool for MCMC methods. Indeed, suppose you built a Markov chain $(X^{(n)})_{n \geq 0}$ admitting π as invariant distribution. Then, under the ergodic theorem hypotheses, if we approximate

⁴That is for all $\varphi \in L^1(\pi)$.

the target distribution π with the corresponding finite- N time average,

$$\int_{\mathcal{X}} \varphi(x) \pi(\mathrm{d}x) \approx \frac{1}{N} \sum_{n=0}^{N-1} \varphi(X^{(n)}), \quad (2.15)$$

we make an error of order $\mathcal{O}(N^{-1/2})$ (cfr. the Central limit theorem).

Remark 2.4. *Note that the l.h.s. of (2.15) is nothing but the expected value of a random variable $\varphi(X)$, where $X \sim \pi$. Since φ is a general function, such Monte Carlo approximation enables the computation of any relevant quantity concerning the target distribution π . For instance, for all events A , setting $\varphi(X) = \mathbb{1}_A(X)$ returns the probability of the event $\pi(A)$; considering $\varphi(X) = X$, we obtain the expected value of a random variable (r.v.) distributed according to π ; taking $\varphi(X) = |X - \mathbb{E}[X]|^2$ gives its variance, and so on and so forth. ♠*

In general, given a probability distribution π , one can build a Monte Carlo approximation by drawing N i.i.d. samples from $X^{(n)} \sim \pi$. Once this ensemble of particles $\{x^{(n)}\}_{n=1}^N$ has been drawn, the **Monte Carlo approximation** of the distribution π

$$\widehat{\pi}_N = \frac{1}{N} \sum_{n=1}^N \delta_{x^{(n)}} \quad (2.16)$$

is available as well. Such approximation is basically an equally weighted sum of Dirac masses centred on the samples, with the convenient property of having a discrete support. In addition, it is a notion of paramount importance for state-of-the-art DA methods. In particular, in the next chapter we will see that particle filtering methods (Section 3.2) extensively exploit the discrete-support property of Monte Carlo approximations.

However, whenever the target distribution π is unknown, it is not possible to directly draw its samples. Markov chain Monte Carlo methods overcome such difficulty by building a Markov chain (from which, the “Markov chain” in MCMC) which is asymptotically distributed as the target distribution, in order to obtain a Monte Carlo approximation $\widehat{\pi}_N$ of π (hence, the “Monte Carlo” in MCMC).

There are many different MCMC methods which can be employed in order to approximate a given target distribution. A notable example is the Gibbs sampler which can be used, for instance, to approximate the joint posterior distribution of the unknown mean and the unknown covariance matrix of a multivariate normal r.v. (see [90, Section 7.4]). However, in the data assimilation community, the most popular MCMC methods are those belonging to the family of Metropolis-Hastings methods.

2.1.1 The Metropolis-Hastings framework

In **Metropolis-Hastings** (MH) algorithms, the Markov Chain Monte Carlo $(X^{(n)})_{n \geq 0}$ which approximates the target distribution π is built according to the following procedure.

Suppose that at time $n - 1$, the chain is in the state $X^{(n-1)} = x$. First, a conditional transition density $q(\hat{x}|x)$ has to be selected. Such density is called the **proposal density** and it characterises the distribution of $\hat{X} \sim q(\cdot|x)$, the proposed new state of the chain at time n . Now, suppose the Markov chain identified by the transition kernel $K(x, d\hat{x}) = q(d\hat{x}|x)$ satisfies the **detailed balance** equation with respect to the target distribution π , i.e.

$$\pi(x)K(x, \hat{x}) = \pi(\hat{x})K(\hat{x}, x). \quad (2.17)$$

Then, since the detailed balance guarantees that the Markov chain with transition kernel $K(x, d\hat{x}) = q(d\hat{x}|x)$ admits π as unique invariant distribution⁵ we constructed our desired MCMC.

Unfortunately, without specific assumptions on the proposal density, q and π are not in detailed balance, and there exist x and \hat{x} in \mathcal{X} such that

$$\pi(x)q(\hat{x}|x) < \pi(\hat{x})q(x|\hat{x}). \quad (2.18)$$

As a consequence, the proposal distribution has to be adjusted in order for (2.17) to hold. In practice, the possibility for a proposed move to be rejected should be included.

To this end, the transition kernel K can be built by introducing the function $m(x, \hat{x})$. For $x \neq \hat{x}$, $m(x, \hat{x})$ is defined as the multiplication of the proposal density $q(\hat{x}|x)$ by the **acceptance density** $a(x, \hat{x})$, the probability of accepting a given proposed move from x to \hat{x} when updating the Markov chain. On the other hand, $m(x, \hat{x})$ is arbitrarily set to zero if $x = \hat{x}$. Then, we consider a transition kernel K of the form

$$K(x, d\hat{x}) = m(x, \hat{x}) d\hat{x} + \left(1 - \int_{\mathcal{X}} m(x, \hat{x}') d\hat{x}'\right) \delta_x(d\hat{x}), \quad (2.19)$$

where $\delta_x(d\hat{x})$ is the Dirac measure defined by $\delta_x(d\hat{x}) = 1$ if $x \in d\hat{x}$ and zero otherwise and, as a consequence, the probability of remaining in state x is $1 - \int_{\mathcal{X}} m(x, \hat{x}') d\hat{x}'$.

Now, let us fix x and \hat{x} with $x \neq \hat{x}$. If (2.18) holds, it means that the detailed balance is broken because the sampling from q results in too many

⁵In fact, integrating (2.17) with respect to \hat{x} yields $\pi(x) = \int \pi(\hat{x})K(\hat{x}, x) d\hat{x}$, that is to say, the distribution π is invariant under the transition kernel K .

moves from \hat{x} to x . Hence, all moves in the opposite direction should be accepted (i.e. $a(x, \hat{x}) = 1$). On the other hand, enforcing (2.17) yields

$$\begin{aligned}\pi(\hat{x})q(x|\hat{x})a(\hat{x}, x) &= \pi(x)q(\hat{x}|x)a(x, \hat{x}) \\ &= \pi(x)q(\hat{x}|x),\end{aligned}$$

from which one obtains $a(\hat{x}, x) = \frac{\pi(x)q(\hat{x}|x)}{\pi(\hat{x})q(x|\hat{x})}$ in case $\pi(\hat{x})q(x|\hat{x}) > 0$. On the contrary, if $\pi(\hat{x})q(x|\hat{x}) = 0$, (2.18) implies $\pi(x)q(\hat{x}|x) = 0$, which means that q is already in detailed balance with π and $a(\hat{x}, x)$ can be set to one. If (2.18) holds with opposite sign, we essentially get the same expression for $a(\hat{x}, x)$.

In conclusion, the resulting probability of accepting a given move from state x to state \hat{x} is⁶

$$a(x, \hat{x}) = 1 \wedge \frac{\pi(\hat{x})q(x|\hat{x})}{\pi(x)q(\hat{x}|x)},$$

where we assume $a(x, \hat{x}) = 1$ if the denominator $\pi(x)q(\hat{x}|x)$ is zero.

To sum up, MH methods work as follows. We choose a proposal distribution $q(d\hat{x}|x)$ and initialise the Markov chain Monte Carlo $(X^{(n)})_{n \geq 0}$ by drawing the initial state $X^{(0)} = x^{(0)}$ from a given initial distribution $p(dx^{(0)})$. Then, at step $n > 0$, draw a proposed new state $\hat{X} = \hat{x}$ from the proposal distribution $q(\cdot|x^{(n-1)})$. Finally, set the new state of the chain according to the acceptance probability, namely

$$X^{(n)} = \begin{cases} \hat{x} & \text{with probability } a(x, \hat{x}) \\ x^{(n-1)} & \text{with probability } 1 - a(x, \hat{x}), \end{cases} \quad (2.20)$$

and then iterate over n .

Of course, there are assumptions on the target distribution π and the proposal distribution $q(d\hat{x}|x)$ that one should check in order to guarantee that the Markov chain Monte Carlo defined above actually converges to π . In brief, the main assumption is that the support of π is contained in the support of the proposal kernel q

$$\text{supp } \pi \subset \bigcup_{x \in \text{supp } \pi} q(\cdot|x). \quad (2.21)$$

For further details we refer to [179, Theorem 6.2.3]

Clearly, each choice of proposal distribution $q(\hat{x}|x)$ results in different algorithms. We now briefly present two subclasses of these algorithms and specify two examples in the case of smoothing distribution estimation.

⁶We adopt the notation \wedge for the minimum operator, namely $x \wedge y := \min\{x, y\}$

Independent samplers

A notable subclass of MH methods is the family of **independent samplers**, whose name stems from the choice of a proposal distribution that is independent on the current state of the chain, i.e. $q(\hat{x}|x) = q(\hat{x})$. In this case the acceptance probability has the form

$$\begin{aligned} a(x, \hat{x}) &= 1 \wedge \frac{\pi(\hat{x})q(x|\hat{x})}{\pi(x)q(\hat{x}|x)} \\ &= 1 \wedge \frac{\pi(\hat{x})q(x)}{\pi(x)q(\hat{x})}. \end{aligned}$$

Let us return to the original smoothing problem for the state-space model defined by (1.1)-(1.2) for a moment. Recall that in this case the target distribution π is the smoothing density $p(z_{0:J}|y_{0:J})$. For the sake of readability, in this description we drop the subscript notation, i.e. we write $z = (z_0, \dots, z_J)$ and $y = (y_0, \dots, y_J)$ instead of $z_{0:J}$ and $y_{0:J}$, respectively.

If we choose the prior computed in (2.1) as symmetric proposal distribution, it follows from (2.7) that

$$\begin{aligned} a(z, \hat{z}) &= 1 \wedge \frac{p(\hat{z}|y)p(z)}{p(z|y)p(\hat{z})} \\ &= 1 \wedge \frac{p(\hat{z}|y)}{p(\hat{z})} \frac{p(z)}{p(z|y)} \\ &= 1 \wedge \frac{p(y|\hat{z})}{p(y|z)} \\ &= 1 \wedge \frac{\prod_{j=0}^J h(y_j|\hat{z}_j)}{\prod_{j=0}^J h(y_j|z_j)}. \end{aligned} \tag{2.22}$$

The quotient in the last line is the ratio between the likelihood of the two points. Note that, since the proposal only takes information from the prior, the accept/reject step is only based on the likelihood: if the likelihood of the proposed move $p(y|\hat{z})$ is larger than the likelihood at the current state $p(y|z)$ the move is accepted with probability one; if the likelihood decreases, the move is accepted with probability smaller than one.

Finally, in the sub-case of Gaussian nonlinear model (Example 1.3), the proposal distribution is given by $q(\hat{z}|z) = q(\hat{z}) \propto \exp\left(-\mathcal{P}(\hat{z})\right)$, where \mathcal{P} is defined in (2.10). The expression for the acceptance probability is then

$$a(z, \hat{z}) = 1 \wedge \exp\left(\mathcal{L}(z) - \mathcal{L}(\hat{z})\right),$$

with \mathcal{L} defined in (2.11). In such Gaussian case, when the proposed likelihood decreases i.e. $\mathcal{L}(\hat{z}) < \mathcal{L}(z)$, we also have that the acceptance probability decays exponentially with the difference of the log-likelihoods.

Random walk Metropolis methods

Another subclass of MH methods is the family of **Metropolis algorithms** [155] which results from choosing a symmetric proposal distribution $q(\hat{x}|x) = q(x|\hat{x})$. As a consequence, the acceptance probability is given by

$$\begin{aligned} a(x, \hat{x}) &= 1 \wedge \frac{\pi(\hat{x})q(x|\hat{x})}{\pi(x)q(\hat{x}|x)} \\ &= 1 \wedge \frac{\pi(\hat{x})}{\pi(x)}. \end{aligned}$$

Example 2.5. In case the state space of the Markov chain is $\mathcal{X} = \mathbb{R}^\ell$, an example of symmetric proposal is $q(\cdot | x) = \mathcal{N}(x, C)$, where C is a positive definite covariance matrix. Such choice results in a proposal \hat{X} that is a Gaussian deviation from the state of the chain at the preceding time step, i.e. $\hat{X} = X^{(n-1)} + D$, where $D \sim \mathcal{N}(0, C)$. Because of this form, Metropolis algorithms are also called random-walk Metropolis methods.

A possible alternative is to choose a deviation variable D that is uniformly distributed on a neighbourhood of the origin, such as $D \sim \mathcal{U}([-c, c]^\ell)$, for some $c > 0$. ♣

Once again, let us specify the acceptance probability in the case of the smoothing problem for the nonlinear state-space model defined in Example 1.3. In such case we have

$$a(z, \hat{z}) = 1 \wedge \exp\left(\mathcal{S}(z; y) - \mathcal{S}(\hat{z}; y)\right),$$

where \mathcal{S} – the neg-log smoothing density – is defined in (2.8). Such expression signifies that, whenever a proposed move increases $\mathcal{S}(\cdot; y)$ the move is accepted with an acceptance probability which decays exponentially with the size of the decrease; on the other hand, it is accepted with probability one any move which decreases the negative-logarithm of the smoothing distribution, i.e. such that $\mathcal{S}(\hat{z}; y) < \mathcal{S}(z; y)$.

Unlike the case of the independence sampler, in case of Metropolis sampler the acceptance probability is influenced by both the prior and the likelihood. On the other hand the proposal step in Metropolis algorithms is nothing but a random walk in the state space. As a consequence, the accept/reject step bears the whole burden of biasing the moves towards maxima of the smoothing density.

More on MCMCs

There are, of course, a number of MCMC methods which can be used in practice, both within and outside the MH class. In addition, there are many technical issues that should be addressed in order to successfully apply MCMC methods to real-life problems, such as the common practice of **burn-in** – throwing the first states of the Markov chain in order to have a Monte Carlo approximation that is independent of its initial distribution $p(x^{(0)})$ – or the **thinning** procedure (extracting a sub-chain $(X^{(kp)})_{k=1}^K$ for $p \in \mathbb{N}$ fixed, so that the resulting particles are uncorrelated). However, going through the details of practical implementation of MCMC goes beyond the scope of this work. We refer to [90] and [179] for further details on this topic.

We now move to introducing the philosophy motivating variational methods, and to illustrate a specific implementation of such approach.

2.2 Variational methods

The MCMC methodology we just introduced extensively resort to Monte Carlo sampling. Unfortunately, whenever the target random variable is defined over a large-dimensional space, sampling a sufficient number of particles can be very demanding from a computational point of view. This is precisely the case for the smoothing problem relative to general state-space model ((1.1) and (1.2)), where the target distribution is the smoothing density $p(z_{0:J}|y_{0:J})$. Here, resorting to MCMC methods means sampling $(J + 1)d_z$ -dimensional, \mathcal{Z}^{J+1} -valued random variables. Whenever there are many observation (i.e. J is large), such space is very large indeed, and Monte Carlo sampling can incur in the so-called **curse of dimensionality** [21, Section 5.16]. For the sake of readability, in this section we drop the subscript notation: whenever it is not differently specified, x stands for $x_{0:J}$ for any variable $x = z, y$.

For such reason, an appealing alternative is to focus on locating a few highly-representative samples of the target distribution, and characterise the target distribution only via these few samples. Typically, the standard approach is to locate the peaks of the target probability density, namely to solving the optimisation problem

$$z^* = \operatorname{argmax}_{z \in \mathcal{Z}^{J+1}} \left\{ p(z|y) \right\}. \quad (2.23)$$

Since this is a problem of locating stationary points of a given function, such techniques are called **variational methods**.

Note that, from a probabilistic point of view, the corresponding maximum points identify the modes of the target distribution, i.e. the states with highest

probability. Besides, in the statistics literature the estimator z^* of the random variable $Z \mid (Y = y)$ defined in (2.23) is called a **maximum a posteriori** (MAP) estimate. Indeed, it is the maximum point of the posterior distribution $p(z|y) \propto p(y|z)p(z)$. As a comparison, consider that other notable statistical estimators include the **conditional mean** $z^* = \mathbb{E}[Z|Y] = \int_{\mathcal{Z}^{J+1}} zp(z|y) dz$, and the **maximum likelihood** (ML) estimate

$$z^* = \operatorname{argmax}_{z \in \mathcal{Z}^{J+1}} p(y|z).$$

Note that, technically speaking the latter is not a Bayesian estimator because it uses no prior distribution in the estimation procedure. Instead, it just maximises the likelihood function $z \mapsto \mathfrak{L}(z; y) = p(y|z)$, for a fixed dataset y . Finally, in the weather forecasting literature, this method is referred to as **4DVAR**, where VAR stands for variational [196, 197]. In fact, models of atmospheric dynamics usually involve three spatial dimensions evolving in time. Hence, solving (2.23) means maximizing a function defined in the four-dimensional spatio-temporal space, from which the 4D in 4DVAR.

Remark 2.6. *Notice that, the argmax function is invariant for positive scaling⁷. As a convenient corollary, such invariance allows one to neglect any normalizing constant in the smoothing distribution. For instance, the Bayes' theorem (2.6) can be invoked and the optimisation problem $\operatorname{argmax}\{p(y|z)p(z)\}$ can be solved instead of $\operatorname{argmax}\{p(z|y)\}$, completely disregarding the normalization constant.* ♠

Recall from (2.8) that the negative-log smoothing density is defined as $\mathcal{S}(z; y) := -\log(p(z|y))$. In light of the last remark, the maximization problem (2.23) can be turned into the equivalent minimization problem

$$z^* = \operatorname{argmin}_{z \in \mathcal{Z}^{J+1}} \left\{ \mathcal{S}(z; y) \right\}. \quad (2.24)$$

Indeed, by definition of $\mathcal{S}(z; y)$ we have that $p(z|y) = \exp(-\mathcal{S}(z; y))$, and the function $x \mapsto \exp(-x)$ is strictly decreasing.

As an example, we now show that in case of linear Gaussian state-space model, the negative-log smoothing density has a single minimum point which can be computed analytically.

⁷i.e. $\operatorname{argmax}\{cf(z)\} = \operatorname{argmax}\{f(z)\}$, for all $c > 0$

Example 2.7 (The Kalman smoother).

Consider the case of a Gaussian linear state-space model as the one defined in Example 1.2. Since in the linear case the deterministic discrete-time map is $g_{j-1}(z_{j-1}) = Mz_{j-1}$, and the deterministic observation operator $H(z_j) = Hz_j$, the negative-log smoothing distribution (2.12) writes

$$\mathcal{S}(z; y) = \frac{1}{2} |z_0 - \mu_0|_{C_0}^2 + \frac{1}{2} \sum_{j=1}^J |z_j - Mz_{j-1}|_{\Sigma_z}^2 + \frac{1}{2} \sum_{j=0}^J |y_j - Hz_j|_{\Gamma}^2.$$

We observe that this is a convex quadratic form in the signal variable $z = (z_0, \dots, z_J) \in \mathbb{R}^{(J+1)d_z}$. In particular, it can be written as

$$\mathcal{S}(z; y) = |z - \mu|_{\Sigma}^2 + c,$$

where $\mu \in \mathbb{R}^{(J+1)d_z}$, Σ is a $((J+1)d_z) \times ((J+1)d_z)$ matrix, and c is a scalar. As a consequence, the unique solution of the minimization problem (2.24) in the linear-Gaussian case is $z = \mu$.

Specifically, the form of the covariance tridiagonal block matrix is

$$\Sigma = \begin{bmatrix} \Sigma_{1,1} & \Sigma_{1,2} & & & & \\ \Sigma_{2,1} & \Sigma_{2,2} & \Sigma_{2,3} & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \Sigma_{J,J+1} & \\ & & & \Sigma_{J+1,J} & \Sigma_{J+1,J+1} & \end{bmatrix},$$

where the $d_z \times d_z$ diagonal blocks are given by $\Sigma_{1,1} = C_0^{-1} + M^T \Sigma_z^{-1} M$, $\Sigma_{jj} = H^T \Gamma^{-1} H + M^T \Sigma_z^{-1} M + \Sigma_z^{-1}$, for $j = 2, \dots, J$, and $\Sigma_{J+1,J+1} = H^T \Gamma^{-1} H + \Sigma_z^{-1}$. For $j = 1, \dots, J$, the upper diagonal blocks have the form $\Sigma_{j,j+1} = -M^T \Sigma_z^{-1}$, whereas the lower diagonal blocks are $\Sigma_{j+1,j} = -\Sigma_z^{-1} M$. On the other hand, the mean $\mu \in \mathbb{R}^{(J+1)d_z}$ is given by $\Sigma \mu = b$, where $b = (b_1, \dots, b_{J+1})$ with $b_1 = C_0^{-1} \mu_0$ and, for $j = 2, \dots, J+1$, $b_j = H^T \Gamma^{-1} y_{j-1}$. Consult [134, Theorem 3.1], for a justification of such expressions.

The method described in this example goes under the name of **Kalman smoother** (KS), and it provides the exact solution of the minimization problem (2.24) in case of Gaussian linear SSM. ♣

We highlight that without specific assumption on the signal and observation model, uniqueness of the solution of (2.23) cannot be guaranteed. Then, the smoothing density has in general multiple maxima and, in principle, one should locate all these maxima to have a good estimator.

In addition, a remarkable property of the Gaussian linear problem discussed in the above example is that the solution of (2.23) can be expressed in closed-form. This is no longer true as soon as the signal model (or the observation model) is nonlinear. In practice, in case of a general state-space model, (2.23) has to be maximised numerically.

Remark 2.8. *There is a vast and mature literature on numerical optimisation, and the choice of a suitable algorithm for a given optimisation problem depends on the properties of the objective function which is being minimised/maximised. For instance, **convexity** of the objective function is usually a desirable property for which a well-established family of classic methods is available [28]. In case of non-convex objective function, further classic optimisation algorithms include **derivative-based algorithms** (e.g., gradient descent/ascent methods, conjugate gradient methods, Newton methods, etc. [163]). In addition, metaheuristic methods which are becoming more and more popular in the last decades include **evolutionary** and **genetic algorithms** (consult [64] and the bibliographical survey [16] for further information), as well as the family of **particle swarm optimisation** methods [62, 168] ♠*

Here, we do not delve further into the details of numerical optimisation methods that can be used to implement 4DVAR. Instead, we just introduce the related path-integral formalism for the nonlinear Gaussian model (Example 1.3) as presented by H. Abarbanel in [12]. As a practical application, we present and discuss the theoretical properties of minAone, the 4DVAR implementation proposed by Abarbanel and collaborators.

2.2.1 The path integral framework

As discussed in Remark 2.4, for a given class of functions φ , integrals of the form

$$\int \varphi(z) \pi(\mathrm{d}z),$$

fully characterise a given probability distribution $\pi(\mathrm{d}z)$. In particular, setting the target distribution to be the posterior smoothing distribution $p(z|y)$ of the Gaussian nonlinear problem defined in Example 1.3, the characterizing integral writes

$$\mathbb{E}[\varphi(Z)|Y = y] = \frac{\int \varphi(z) \exp\left(-\mathcal{S}(z; y)\right) \mathrm{d}z}{\int \exp\left(-\mathcal{S}(z; y)\right) \mathrm{d}z}. \quad (2.25)$$

We recall that such integrals are over the signal process **path space** \mathcal{Z}^{J+1} , as we are employing the notation $z = z_{0:J}$, so that $dz = dz_0 \cdots dz_J$. This is the reason why [12] refers to integral (2.25) as the **path integral**. In addition, $\mathcal{S}(z; y)$ is called the **action**, in analogy to the quantum-physics terminology.

As we discussed in the previous section, the aim is to approximate the action $\mathcal{S}(z; y)$ in order to obtain an approximation of the smoothing distribution $p(z|y) \propto \exp(-\mathcal{S}(z; y))$. In Example 2.2, we showed that $\mathcal{S}(z; y)$, the negative-log smoothing density, has the form

$$\mathcal{S}(z_{0:J}; y_{0:J}) = \frac{1}{2} |z_0 - \mu_0|_{C_0}^2 + \frac{1}{2} \sum_{j=1}^J |z_j - g_{j-1}(z_{j-1})|_{\Sigma_z}^2 + \frac{1}{2} \sum_{j=0}^J |y_j - H(z_j)|_{\Gamma}^2.$$

In what follows, we also assume that the covariance matrices are all diagonal, i.e. $\Sigma_z = \sigma^2 I_{d_z}$, and $\Gamma = \gamma^2 I_{d_y}$. In addition, we consider that the observation operator is the linear projection on the first d_y components of the signal $z_j \in \mathcal{Z}$ (it could actually be any projection on d_y components, not necessarily the first ones). In practice, H is assumed to be linear, i.e. $H(z_j) = Hz_j$ where in the r.h.s H is a block $d_y \times d_z$ matrix $H = [I_{d_z}, 0]$. Hence, we obtain

$$\mathcal{S}(z; y) = \overbrace{\frac{1}{2\sigma^2} \sum_{j=0}^J |z_j - g_{j-1}(z_{j-1})|^2}^{\mathcal{P}(z)} + \overbrace{\frac{1}{2\gamma^2} \sum_{j=0}^J |y_j - Hz_j|^2}^{\mathcal{L}(z;y)}, \quad (2.26)$$

where we set $g_0(z_{-1}) \equiv \mu_0$ and $C_0 = \Sigma_z$ in order to simplify the problem. In such way, the negative-log prior $\mathcal{P}(z)$ has both the initial condition and the time-evolution term pooled in a single sum.

Equation 2.26 highlights that the process of minimizing $\mathcal{S}(z; y)$ involves a trade-off between minimizing the negative-log prior density $\mathcal{P}(z)$ (responsible for the dynamical evolution) and the minimization of the negative-log likelihood $\mathcal{L}(z)$ (responsible for the data-fitting). In fact, we have a sort of a weighted sum, in which the two weights $\frac{1}{\sigma^2}$ and $\frac{1}{\gamma^2}$ determine the relative importance of the prior, and the likelihood term, respectively. To further simplify the notation, in what follows we omit the dependence of the negative-log smoothing distribution on the data y , i.e. we write $\mathcal{S}(z)$ instead of $\mathcal{S}(z; y)$.

We now introduce the algorithm minAone, which was first presented in [12] and then improved and discussed in [219, 220]. We do so in a formal manner, in the sense that we do not state explicitly the hypothesis one should make on $\mathcal{S}(z)$ in order to justify all the computation we perform.

2.2.2 minAone

As in 4DVAR, the core idea of minAone is to locate the global minimum of the action $\mathcal{S}(z)$. However, as a consequence of possibly high nonlinearity in the discrete-time map $g_{j-1}(z_{j-1})$, the action $\mathcal{S}(z)$ may have many local minimums $(z^{(0)}, z^{(1)}, z^{(2)}, \dots)$. This implies that a direct minimization of the action might be very sensitive to the initial guess in the optimisation algorithm.

We here assume that the minimizing paths $(z^{(i)})_i$ are sorted increasingly with respect to the action values, i.e.

$$\mathcal{S}(z^{(0)}) < \mathcal{S}(z^{(1)}) < \dots \quad (2.27)$$

With this specification, $z^{(0)}$ is the global minimum point of the action, $z^{(1)}$ is the second smallest local minimum etc.

In minAone, the problem of having many local minimums is overcome by considering the dynamical model error σ^2 as a scalable parameter. In general, a large σ^2 value would reflect the belief that the signal model is inaccurate. On the other hand, a small σ^2 value represents a small dynamical noise, and it shows a good confidence in the signal model.

In fact, if we set $\frac{1}{\sigma^2} = 0$ (corresponding to a infinitely inaccurate signal model, $\sigma^2 \rightarrow \infty$), there is no influence of $\mathcal{P}(z)$ on the action, which is only given by the likelihood term, i.e. $\mathcal{S}(z) = \frac{1}{2} \sum_{j=0}^J \left(\frac{|y_j - Hz_j|}{\gamma} \right)^2$. Hence, in case $\frac{1}{\sigma^2} = 0$, all the minimum points of the action $z^{(i)} = (z_0^{(i)}, \dots, z_J^{(i)})$ have the first d_y components equal to those of y , while the remaining $d_z - d_y$ components are free parameters. As $\frac{1}{\sigma^2}$ increases (or equivalently σ^2 decreases), the relative weight of the prior signal-model part of the action increases too. As a consequence the nonlinearity of the problem manifests, and many local minimums arise.

The proposal of minAone is to consider

$$\frac{1}{\sigma^2} = R_{f0} \alpha^\beta,$$

for $\alpha > 1$, $\beta \geq 0$ and small $R_{f0} < 1$, and then, solve the minimization problem

$$\operatorname{argmin}_{z \in \mathbb{R}^{(J+1)d_z}} \left\{ \frac{R_{f0} \alpha^\beta}{2} \sum_{j=0}^J |z_j - g_{j-1}(z_{j-1})|^2 + \frac{1}{2\gamma^2} \sum_{j=0}^J |y_j - Hz_j|^2 \right\}, \quad (\beta\text{-prob})$$

in an iterative manner, for $\beta = 0, 1, 2, \dots, \beta_{max}$.

In practice, suppose that we solved (β -prob) for $\beta = \beta_1$ with N independent runs of a suitable iterative numerical optimisation algorithm, obtaining a series of solutions $z_{\beta_1}^{(i)}$, for $i = 1, \dots, N$. Then, the i -th instance of (β -prob) for

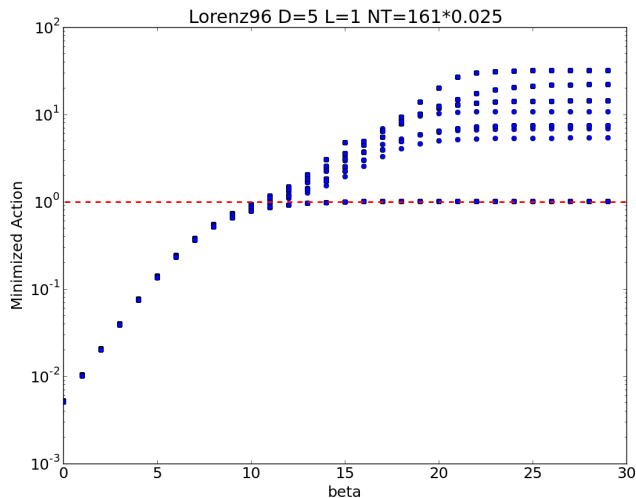


Figure 2.1: Courtesy of Jingxin Ye. Evolution of $N = 100$ minimums of the action of the Lorenz96 Gaussian problem for $\alpha = 2$ $\beta = 0, \dots, \beta_{max} = 29$. The red dashed lines represent the confidence interval of the χ^2 distribution. Source: minAone build pack, available at the web page [2].

$\beta = \beta_1 + 1$ is initialised with $z_{\beta_1}^{(i)}$. In this way we track the evolution of the action minimums found for $\frac{1}{\sigma^2} \approx 0$, by gradually increasing $\frac{1}{\sigma^2}$ and using at each step a consistent initial guess. Eventually, when the precision $\frac{1}{\sigma^2}$ is very large, the signal model is essentially deterministic.

In [219], it was empirically shown that, in many examples, the cloud of N minimal action values in the $\beta - \mathcal{S}$ plane often presents a bifurcation in the large- β region. As shown in Figure 2.1, a large number of local minimums gradually moves away from a single absolute minimum. This means that the prior part of the action $\frac{1}{2} \sum_{j=0}^J \left(\frac{|z_j - g_{j-1}(z_{j-1})|}{\sigma} \right)^2$ is zero independently of σ , which can only be explained if the minimizing path – corresponding to the absolute minimum – fundamentally matches the model dynamics.

Such observation also allows one to give an estimate of the action absolute minimum. Indeed, $\mathcal{P}(z) \approx 0$ means that the action value is only given by the remaining likelihood part, i.e. $\mathcal{S}(z) \approx \frac{1}{2} \sum_{j=0}^J \left(\frac{|y_j - Hz_j|}{\gamma} \right)^2$. Since the normalised random variable $\frac{y_j - Hz_j}{\gamma}$ follows a standard d_y -dimensional normal distribution $\mathcal{N}(0, I_{d_y})$, then $\left(\frac{|y_j - Hz_j|}{\gamma} \right)^2$ follows a $\chi^2(d_y)$ -distribution, and $\sum_{j=0}^J \left(\frac{|y_j - Hz_j|}{\gamma} \right)^2 \sim \frac{1}{2} \chi^2((J+1)d_y)$, a scaled chi squared distribution with $(J+1)d_y$ degrees of freedom. The latter r.v. has mean $\frac{(J+1)d_y}{2}$ and stan-

dard deviation $\sqrt{(J+1)d_y}$, so that the absolute minimum lays in the interval $\left[\frac{(J+1)d_y}{2} - \sqrt{(J+1)d_y}, \frac{(J+1)d_y}{2} + \sqrt{(J+1)d_y}\right]$ with high probability⁸.

On the contrary, if in the large- β regime one finds an absolute minimum which is far from the confidence interval, then it is a hint that the signal model does not capture the data dynamics.

The actual Python source code for minAone is available at Jingxin Ye's github web page [2]. Such implementation makes use of the interior point optimiser IPOPT [211], which is publicly available at the web page [3]. Also, a series of Python scripts implemented by Bryan Toth are employed in order to automatically build the action function of a given differential model of the signal, along with its partial derivatives [198].

An overview of the methods introduced in this chapter is illustrated in the r.h.s. branch of the mind map in Figure 2.2. In the next chapter, we introduce the sequential methodology characterizing filtering algorithms and detail the methods mentioned in the leaves of the l.h.s. branch.

⁸Actually, a more reliable interval can be given by considering the 95%-confidence interval of the χ^2 -distribution with $(J+1)d_y$ degrees of freedom. To compute it, one can resort to the cumulative distribution function of the χ^2 distribution and the corresponding tables.

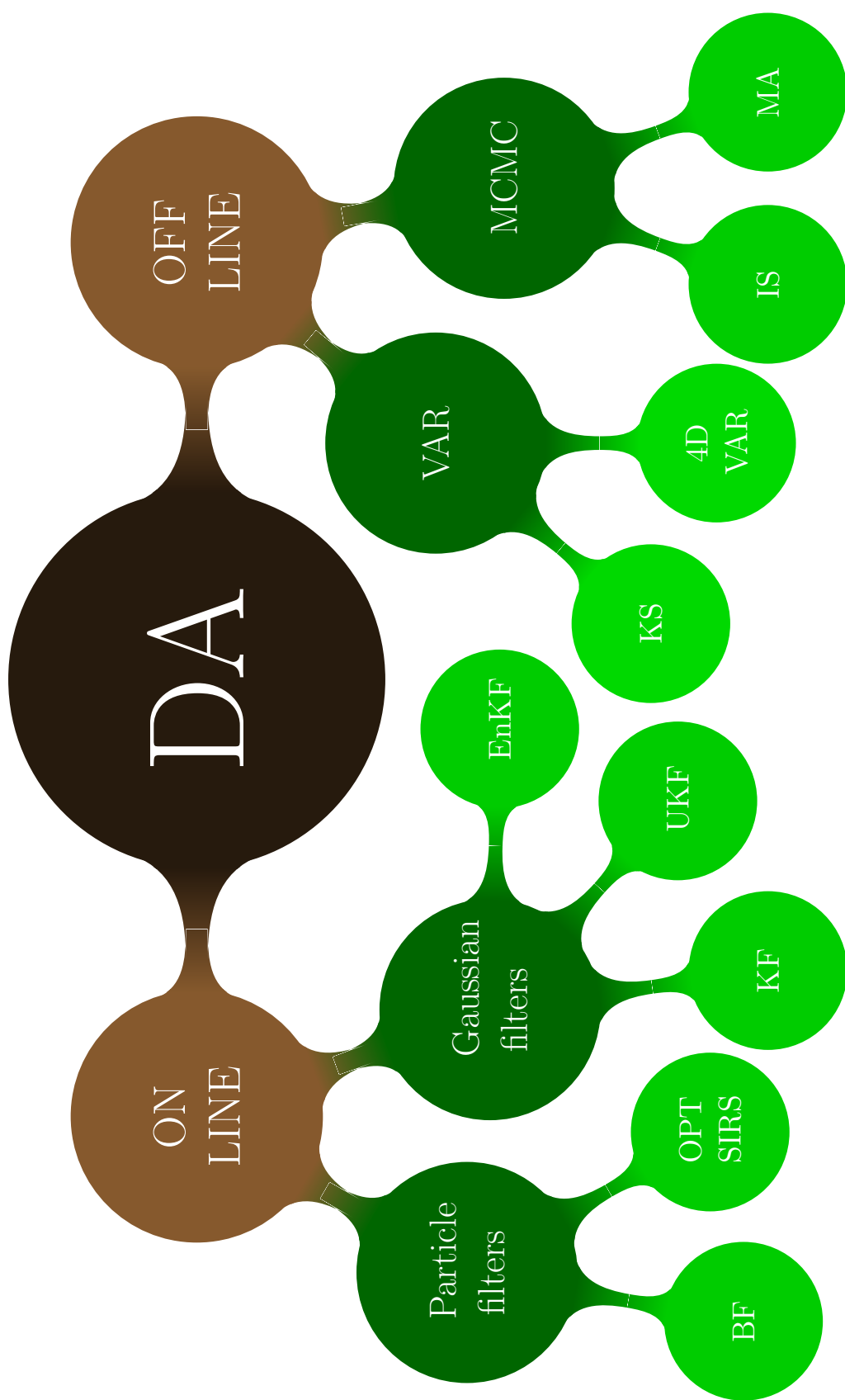


Figure 2.2: Mindmap of data assimilation algorithms. Legend: DA = data assimilation, BF=bootstrap filter, OPT-SIRS=optimal importance resampling, KF=Kalman filter, UKF=unscented Kalman filter, EnKF=ensemble Kalman filter, VAR=variational, KS=Kalman smoother, MCMC=Markov chain Monte Carlo, IS=independence sampler, MA=Metropolis algorithms.

Chapter 3

On-line filtering methods

In the previous chapter, we presented and discussed smoothing algorithms, which we recall are methods that aim at computing or (more often) approximating the smoothing distribution $p(z_{0:J}|y_{0:J})$. In such discussion, we noted that these are off-line methods in the following sense. Suppose a smoothing algorithm has already been run in the data assimilation time window $\{0, \dots, J\}$, and it returned the approximated smoothing distribution $\hat{p}(z_{0:J}|y_{0:J})$. Then, suppose a new measurement y_{J+1} becomes available, and we thus want to estimate the smoothing distribution in the extended time window $\{0, \dots, J+1\}$. The issue here is that it is not possible to estimate the “extended” smoothing distribution $p(z_{0:J+1}|y_{0:J+1})$ by simply incorporating the previous estimation of $p(z_{0:J}|y_{0:J})$ in a further step of the smoothing algorithm. Fortunately, another class of sequential methods comes in handy in case an on-line version of DA is desired: the family of filtering methods.

Indeed, filtering algorithms aim at approximating the **filtering distribution**

$$\left(p(z_j|y_{0:j}) \right)_{j=0}^J = \left(p(z_0|y_0), p(z_1|y_{0:1}), \dots, p(z_J|y_{0:J}) \right)$$

in a sequential fashion. Namely, at each time j the filtering distribution $p(z_j|y_{0:j})$ is updated only using the information contained in $p(z_{j-1}|y_{0:j-1})$, the filtering distribution at the preceding time step $j-1$.

The filtering update is typically performed applying a two-step procedure, which is graphically outlined in Figure 3.1. First, in the **prediction step**, the filtering distribution is pushed forward in time only relying on the signal model, i.e. applying the functional mapping

$$p(z_j|y_{0:j-1}) = \int_{\mathcal{Z}} p(z_j|z_{j-1})p(z_{j-1}|y_{0:j-1}) dz_{j-1}. \quad (3.1)$$

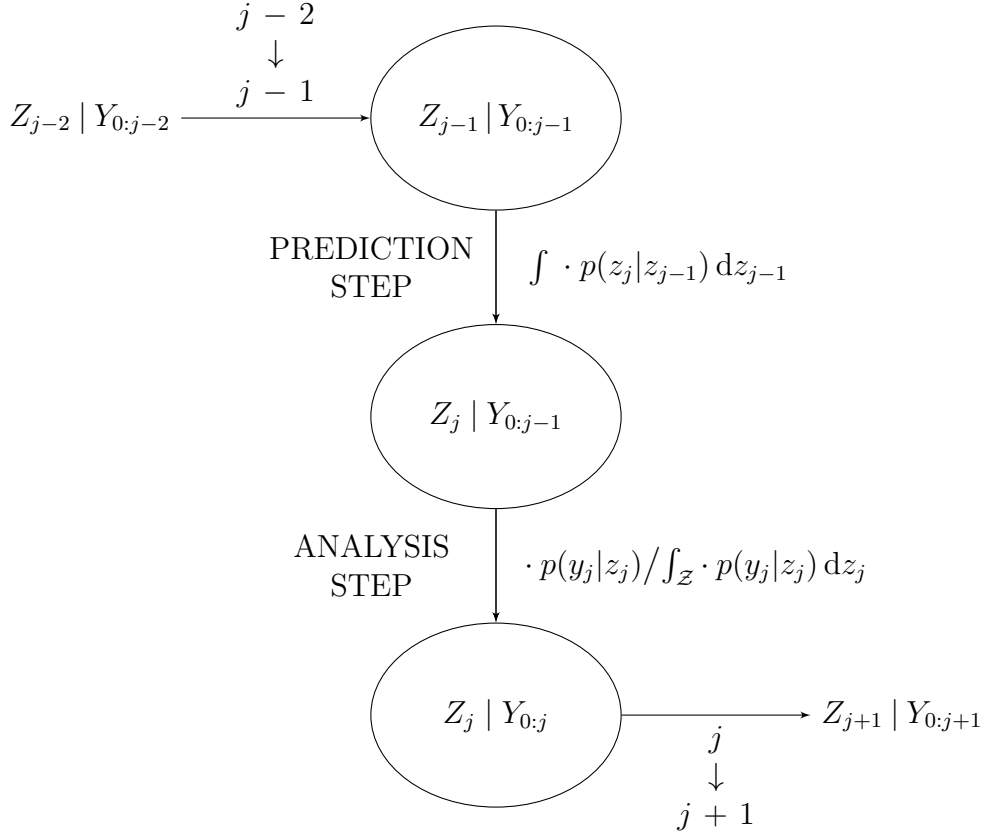


Figure 3.1: Graphical representation of the prediction-analysis procedure for the filtering update from time step $j - 1$ to time j . On the right hand side of the downward arrows, the functional mapping of the density of the r.v. in the circle above into the density of the r.v. in the circle below is reported.

The resulting **predicted distribution** $p(z_j|y_{0:j-1})$ is a sort of prior distribution for the state of the system at time j , and it does not take the j -th observation y_j into account. In the subsequent **analysis step**, the filtering distribution $p(z_j|y_{0:j})$ is computed multiplying the predicted distribution by the likelihood

$$p(z_j|y_{0:j}) \propto p(y_j|z_j)p(z_j|y_{0:j-1}). \quad (3.2)$$

according to Bayes' theorem. These two steps run iteratively.

Note that the prediction step is a linear operator over the space of filtering densities at time $j - 1$, i.e. the space of $p(z_{j-1}|y_{0:j-1})$'s. On the contrary, the analysis step maps $p(z_j|y_{0:j-1})$ to $p(z_j|y_{0:j})$ in a nonlinear way. In fact, the normalization constant omitted in (3.2) is given by an integral sum which involves the predicted density – specifically, $c = \int_{\mathcal{Z}} p(z_j|y_{0:j-1})p(y_j|z_j) dz_j -$

causing the nonlinearity.

In what follows we describe in detail three filtering algorithms: the ensemble Kalman filter (EnKF), the bootstrap filter (BF), and the optimal importance resampling (OPT-SIRS). Note that both BF and OPT-SIRS fall into the broader family of particle filters (presented in Section 3.2), whereas the EnKF is one of those methods which strongly rely on the Gaussianity of the state-space model, and which we refer to as “approximate Gaussian filters” (see the following Section 3.1). As a consequence, for each of these two methodologies we first give a brief overview of the method, and then we instantiate the general case with the above mentioned (and more) algorithms.

Unless specified otherwise, we refer to [134, Chapter 4] and references therein for rigorous proofs all mathematical statements of this chapter.

3.1 Approximate Gaussian filters

Loosely speaking, approximate Gaussian filters can be identified as those methods that generalise the Kalman filter (KF) [107]. This is an exact algorithm which results from sequentially solving the filtering problem in a analytical manner, in case the state-space model is both linear and Gaussian, as in Example 1.2. However, Kalman-like filters can be applied to nonlinear state-space model if the predicted distribution is assumed to be Gaussian whose predicted mean and covariance matrix are approximated in some consistent way. Nevertheless, Note that the state space model should still be the whole real space $\mathcal{Z} = \mathbb{R}^{d_z}$ and, in particular, it still needs to be Gaussian as in Example 1.3.

Now, before introducing the EnKF (Section 3.1.3), we first present the complete KF algorithm and some first generalizations.

3.1.1 The Kalman filter

Consider a signal model defined for $j \in \{0, \dots, J\}$ of the form

$$\begin{cases} Z_0 \sim \mathcal{N}(\mu_0, C_0) \\ Z_j = MZ_{j-1} + \epsilon_j, \end{cases}$$

which is coupled to the observation model

$$Y_j = HZ_j + \varepsilon_j.$$

The mutually independent noise processes $(\epsilon_j)_{j>0}$ and $(\varepsilon_j)_{j\geq 0}$ are i.i.d. sequences with $\epsilon_j \sim \mathcal{N}(0, \Sigma_z)$ and $\varepsilon_j \sim \mathcal{N}(0, \Gamma)$.

Algorithm 3.2 Kalman filter (KF)

```
1: function KF( $[\mu_0, C_0], [M, \Sigma_z], [H, \Gamma], y_{0:J}$  )
2:   for  $j = 1 : J$  do
3:      $\hat{\mu}_j \leftarrow M\mu_{j-1}$ 
4:      $\hat{C}_j \leftarrow MC_jM^T + \Sigma_z$  } # Prediction
5:      $S_j \leftarrow H\hat{C}_jH^T + \Gamma$ 
6:      $K_j \leftarrow \hat{C}_jH^TS_j^{-1}$ 
7:      $\delta_j \leftarrow y_j - H\hat{\mu}_j$ 
8:      $\mu_j \leftarrow \hat{\mu}_j + K_j\delta_j$  } # Analysis
9:      $C_j = (I - K_jH)\hat{C}_j$ 
   return  $\mu_{0:J}, C_{0:J}$ 
```

Since both the signal and data models are Gaussian and linear, we obtain a state-space model which is identical to the one defined in Example 1.2. Under such assumptions, it can be proved that the solution to the associated filtering problem is a sequence of Gaussian probability distributions $p(z_j|y_{0:j}) = \mathcal{N}(\mu_j, C_j)$, for $j \in \{0, \dots, J\}$. The main justification is that linear mappings of normal random variables preserve Gaussianity.

As a consequence, solving the filtering problem boils down to computing the mean μ_j and the covariance C_j of all $p(z_j|y_{0:j})$'s. The exact scheme to so is reported in Algorithm 3.2, where we observe that in the prediction step, the mean is pushed forward in time solely relying on the signal model (line 3). On the other hand, the analysis update of the filtering mean and covariance for the Kalman filter relies on both the **Kalman gain** K_j and the **innovation** δ_j . The former subsumes the data model (notice the presence of Γ and H at lines 5-6), whereas the latter measures how distant the predicted mean $\hat{\mu}_j$ is to the j -th observation y_j (line 7), and then adjusts the predicted mean and covariance accordingly (lines 8-9). Finally, the algorithm returns the filtering mean and covariance which fully characterise the exact filtering distribution in case of a linear SSM. See Figure 3.3 for a graphical representation of the prediction-analysis procedure in case of Kalman filter.

Remark 3.1. *Note that the main assumption needed to guarantee that the KF provides the exact filtering distribution, is that the covariance matrices C_0 , Σ_z , and Γ are positive definite. In case of diagonal covariance matrices, this means that all components of the initial condition, of the signal variable, and of the data variable should have non-zero variance. ♠*

Unfortunately, in real life very few phenomenon can be described by linear models. As a consequence the Kalman filter, although exact, is often of little

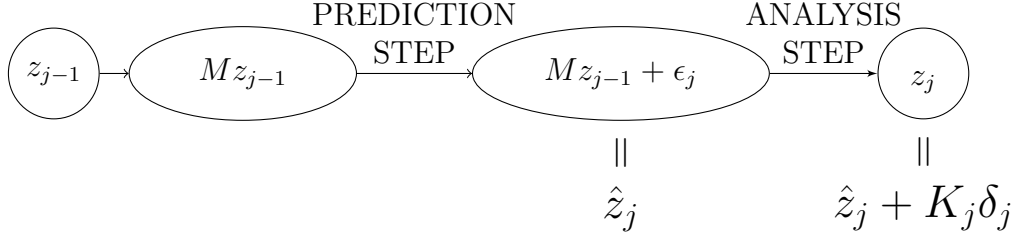


Figure 3.3: Graphical representation of the Kalman filter. The innovation δ_j stands for $y_j - H\hat{z}_j$.

practical use. Thus, the question of how to generalise the Kalman filter to nonlinear (but still Gaussian) signal and data models arises. We address this topic in the subsequent two sections.

3.1.2 The extended and the unscented Kalman filter

The simple remark that the Kalman mean update (line 8 in Algorithm 3.2) essentially stems from a minimization principle, allows one to generalise the Kalman update to the nonlinear case. In fact, in case of Gaussian nonlinear SSM (refer to (1.5)-(1.6) in Example 1.3) we can define an equivalent of the Kalman mean update through

$$\mu_j = \operatorname{argmin}_{z_j \in \mathcal{Z}} \mathcal{F}_j(z_j; y_j), \quad (3.3)$$

with

$$\mathcal{F}_j(z_j; y_j) = \frac{1}{2} |z - \hat{\mu}_j|_{\hat{C}_j}^2 + \frac{1}{2} |y_j - H(z_j)|_{\Gamma}^2.$$

Note that, with such definition (3.3) returns the exact Kalman filter mean and covariance update in the linear case.

On the other hand, in case of nonlinear SSM, the definition of the predicted mean can be straightforwardly set to the equivalent of the linear case, namely $\hat{\mu}_j = g_{j-1}(z_{j-1})$ or in some variants $\hat{\mu}_j = g_{j-1}(z_{j-1}) + \epsilon_j$ for some sample of the noise random variable ϵ_j . On the contrary, there is no direct analogous of the KF predicted covariance matrix in the nonlinear case. Thus, the choice of \hat{C}_j is somewhat arbitrary and is what differentiates the different approximate Gaussian filters.

For instance, if the deterministic signal update function g_{j-1} is differentiable, setting $M_{j-1} = \frac{dg_{j-1}}{dz}(\mu_{j-1})$ (i.e., M_{j-1} is the Jacobian matrix of g_{j-1}) and $\hat{C}_j = M_{j-1}C_{j-1}M_{j-1}^T + \Sigma_z$ results in the **extended Kalman filter** (ExKF). This choice is essentially equivalent to linearizing the signal model in the vicinity of the current estimate in the prediction step.

Also, an even simpler choice stems from setting $\hat{C}_j \equiv \hat{C}$, which generates the so-called **3DVAR** algorithm [143, 142]. Note the analogy with the smoothing algorithm 4DVAR, which simply applies the same variational principle to the batch signal variable $z_{0:j}$ (cfr. Section 2.2).

Remark 3.2. *Note, that the variational formulation (3.3) essentially corresponds to postulate that the predicted distribution $p(z_j|y_{0:j-1})$ is Gaussian with mean $\hat{\mu}_j$ and covariance matrix \hat{C}_j , which is in general not true for nonlinear state-space models.* ♠

An alternative which relies on the concept of unscented transformation [103], is to deterministically locate a minimal set of weighted **sigma points** around the mean μ_{j-1} which are specifically designed to match the first and second moments of the filtering p.d.f at time $(j-1)$. This results in the so-called **unscented Kalman filter** (UKF) which was proposed by J. Uhlmann and S. Julier in 1997 [104]. The UKF is reported to yield better estimation performances than the ExKF in many applications where the SSM is highly nonlinear in the observations time scale. Note that strictly speaking, the UKF does not apply the above-stated variational principle, but it still generalises the KF to nonlinear problems using a Kalman-like analysis update. We recommend [190, Section 14.3] for a clear description of the UKF algorithm in the additive-noise case which adopts a notation compatible to the one used in this text. Also, consult [153] for a recent review and classification of the vast plethora of UKF variants.

We now move to a further approximate Gaussian filter which relies on the variational principle (3.3) by introducing a different form of \hat{C}_j .

3.1.3 The ensemble Kalman filter

The EnKF is an approximate Gaussian filter, which generalises the linear Kalman filter by introducing a Monte Carlo approximation for the filtering random variable $Z_j|Y_{0:j}$. It was proposed by G. Evensen in [67, 68] and then refined in [32]. Since then, it has been successfully applied in many practical domains (see for instance [115, 175, 205] and references therein) and used as a benchmark to test different Bayesian DA methods [41, 84].

In the prediction step, the EnKF draws $N \in \mathbb{N}$ independent samples of the signal model and uses them to build the predicted ensemble $\{\hat{z}_j^{(n)}\}_{n=1}^N$. Then, the empirical mean $\hat{\mu}_j = \frac{1}{N} \sum_{n=1}^N \hat{z}_j^{(n)}$ and the empirical covariance matrix

$$\hat{C}_j = \frac{1}{N-1} \sum_{n=1}^N (\hat{z}_j^{(n)} - \hat{\mu}_j)(\hat{z}_j^{(n)} - \hat{\mu}_j)^T \quad (3.4)$$

Algorithm 3.4 Ensemble Kalman filter (EnKF)

```

1: function ENKF( $N, [\mu_0, C_0], [g_{0:J}(\cdot), \Sigma_z], [H, \Gamma], y_{0:J}$ )
2:   draw  $z_0^{(n)} \sim \mathcal{N}(\mu_0, C_0), \quad \forall n = 1 : N$            # Initialization
3:   for  $j = 1 : J$  do
4:     draw  $\hat{z}_j^{(n)} \sim \mathcal{N}(g_{j-1}(z_{j-1}^{(n)}), \Sigma_z), \quad \forall n = 1 : N$ 
5:      $\hat{\mu}_j \leftarrow \frac{1}{N} \sum_{n=1}^N \hat{z}_j^{(n)}$ 
6:      $\hat{C}_j \leftarrow \frac{1}{N-1} \sum_{n=1}^N (\hat{z}_j^{(n)} - \hat{\mu}_j)(\hat{z}_j^{(n)} - \hat{\mu}_j)^T$ 
7:      $S_j \leftarrow H\hat{C}_jH^T + \Gamma$ 
8:      $K_j \leftarrow \hat{C}_jH^TS_j^{-1}$ 
9:     draw  $y_j^{(n)} \sim \mathcal{N}(y_j, \Gamma), \quad \forall n = 1 : N$ 
10:     $\delta_j^{(n)} \leftarrow y_j^{(n)} - H\hat{z}_j^{(n)}, \quad \forall n$ 
11:     $z_j^{(n)} \leftarrow \hat{z}_j^{(n)} + K_j\delta_j^{(n)}, \quad \forall n$ 
   return  $\left\{ z_{0:J}^{(n)} \right\}_{n=1}^N$ 

```

} # Prediction

} # Analysis

are used to update the filtering ensemble $\{z_j^{(n)}\}_{n=1}^N$ by implementing a KF-like analysis step.

The complete EnKF procedure is as follows.

Initialise Draw the initial ensemble by independently sampling N particles from the initial distribution $\mathcal{N}(\mu_0, C_0)$.

Then, for time steps $j = 1, \dots, J$ apply:

Prediction Build the predicted ensemble by drawing N independent samples from the signal model (1.5). Then, compute the empirical covariance matrix of the resulting ensemble according to (3.4).

Analysis Compute the Kalman-gain matrix K_j . Then, update the filtering ensemble by correcting the predicted ensemble according to $K_j\delta_j^{(n)}$, where $\delta_j^{(n)} = y_j - H\hat{z}_j^{(n)}$ is the n -th innovation.

The details of such procedure are given in Algorithm 3.4, where the pseudo-code for the ensemble Kalman filter is given. Note that the expressions for S_j (line 7), for the Kalman gain K_j (line 8), and for the innovation δ_j (line 10) are only valid for linear observation model, i.e. such that deterministic observation operator H is a $d_y \times d_z$ matrix i.e. $H(z) = Hz$.

After it was proposed, many efforts have been done in order to theoretically justify the EnKF. Unfortunately, to the best of our knowledge, most of the work has been done only to prove the convergence of the EnKF in the large

ensemble limit (i.e., for $N \rightarrow \infty$) [136], recovering the Kalman filter in case of linear models. However, more recent works also focus on the EnKF stability in the more practical conditions where the ensemble size N is fixed and possibly small [114, 184].

Note that in practical applications, method-parameter fine-tuning is a key (and time-consuming) issue necessary for the success of EnKF application. See Section 9.1.3 for an example in the context of single-neuron model parameter estimation. In addition, both covariance inflation and localization [69, 109] can be effective to prevent filter divergence [17, 79].

In the next section, we introduce and instantiate a different ensemble-based methodology which is becoming more and more popular in the last decades due to its flexibility and its good theoretical properties.

3.2 The particle filter

Dating back to at least [78], **particle filters** (PF) are algorithms which introduce an ensemble of particles $\{z_j^{(n)}\}_{n=1}^N$ in order to approximate the filtering distribution $p(z_j|y_{0:j})$. Analogously to the Monte Carlo sampling strategy described in Section 2.1, the **ensemble** $\{z_j^{(n)}\}_{n=1}^N$ is a set of i.i.d. realizations of the random variable $Z_j \mid (Y_{0:j} = y_{0:j})$ which allows one to approximate the filtering distribution $p(z_j|y_{0:j})$ with its **Monte Carlo approximation**

$$\hat{p}_N(z_j|y_{0:j}) := \sum_{n=1}^N w_j^{(n)} \delta_{z_j^{(n)}}(z_j). \quad (3.5)$$

In the above equation $N \in \mathbb{N}$ is the **ensemble size**, whereas $\delta_{z_j^{(n)}}(\cdot)$ stands for the Dirac delta function centred on the n -th sample $z_j^{(n)}$. Besides, $w_j^{(n)}$ is the **importance weight** which quantifies the probability that the random variable $Z_j \mid (Y_{0:j} = y_{0:j})$ assumes the value $z_j^{(n)}$. Note that particle filters are also called **sequential Monte Carlo (SMC) methods**. In fact, (3.5) is a nothing but a non-uniformly weighted version of the Monte Carlo approximation introduced in (2.16) when presenting the MCMC methodology.

In general, all sequential Monte Carlo methods are structured as follows. In the prediction step, the ensemble particles $z_{j-1}^{(n)}$ are pushed forward in time by sampling the **importance sampling distribution** $q(z_j|z_{j-1}^{(n)}, y_{0:j})$ (also known as **proposal distribution**). This generates the **proposed ensemble** $\{\hat{z}_j^{(n)}\}_{n=1}^N$, which is also referred to as predicted ensemble. In the analysis step, the importance weights are then updated according to their likeliness by

Algorithm 3.5 Bootstrap filter (BF)

```
1: function BF( $N, f_0(\cdot), f_{0:J}(\cdot|\cdot), h(\cdot|\cdot), y_{0:J}$ )
2:   draw  $z_0^{(n)} \sim f_0, \quad \forall n = 1 : N$            # Initialization
3:   for  $j = 1 : J$  do
4:     draw  $\hat{z}_j^{(n)} \sim f_{j-1}(\cdot | z_{j-1}^{(n)}), \quad \forall n = 1 : N$    # Prediction
5:      $\hat{w}_j^{(n)} \leftarrow h(y_j | \hat{z}_j^{(n)})$ 
6:     normalize  $\{\hat{w}_j^{(n)}\}$  } # Analysis
7:     draw  $z_j^{(n)} \sim \sum_{n=1}^N \hat{w}_j^{(n)} \delta_{\hat{z}_j^{(n)}}, \quad \forall n = 1 : N$    # Resampling
   return  $\left\{ z_{0:J}^{(n)} \right\}_{n=1}^N$ 
```

applying the Bayes' theorem (3.2). Note the analogies with the Metropolis-Hastings methodology described in Section 2.1.1.

This simple procedure generates a wide class of methods whose only difference is the choice of the proposal distribution q . It should be emphasised that, unlike the EnKF, particle filters apply to the general state-space model defined in (1.1) and (1.2), with no need for Gaussianity assumptions. This is one of the reasons of their recent and growing popularity.

In the rest of this section, we present in detail the algorithms for two instances of particle filters: the bootstrap filter and the optimal sequential importance resampling. Then, we mention some other sequential Monte Carlo method based on Gaussian-like proposal distributions in Section 3.2.3, and finally conclude the chapter by reporting some results concerning the well-posedness and consistency of particle filter methods (Section 3.2.4).

3.2.1 The bootstrap filter

The **bootstrap filter** is a particular version of the particle filter, which results from choosing the signal model density (1.3) as proposal distribution, namely

$$q(z_j | z_{j-1}^{(n)}, y_{0:j}) = p(z_j | z_{j-1}^{(n)}).$$

Such choice generates the following algorithm.

Initialise Initialise the BF filtering distribution, i.e. for $n = 1, \dots, N$, draw the initial particles $z_0^{(n)} \sim p(z_0)$. Then, set the initial approximate distribution to be $\hat{p}_N(z_0) = \sum_{n=1}^N \frac{1}{N} \delta_{z_0^{(n)}}(z_0)$.

Then, for $j = 1, \dots, J$ apply:

Prediction Build the predicted ensemble by sampling the proposal distribution which, for the bootstrap filter, is the signal model distribution (1.3)

$$\hat{z}_j^{(n)} \sim p(z_j | z_{j-1}^{(n)})$$

Analysis Compute the ensemble weights according to the likelihood

$$\hat{w}_j^{(n)} = p(y_j | \hat{z}_j^{(n)}) / \sum_{n=1}^N p(y_j | \hat{z}_j^{(n)})$$

Then, set the predicted filtering density at time j to be $\sum_{n=1}^N \hat{w}_j^{(n)} \delta_{\hat{z}_j^{(n)}}(\hat{z}_j)$

Resample Resample the predicted filtering distribution in order to obtain the filtering ensemble $\{z_j^{(n)}\}_{n=1}^N \sim \sum_{n=1}^N \hat{w}_j^{(n)} \delta_{\hat{z}_j^{(n)}}$. As a consequence, the filtering distribution approximation at time j has uniform weights, i.e.

$$\hat{p}_N(z_j | y_{0:j}) = \frac{1}{N} \sum_{n=1}^N \delta_{z_j^{(n)}}(z_j).$$

Algorithm 3.5 summarises the BF algorithm in case of general state-space model as defined by (1.1) and (1.2).

3.2.2 The optimal sequential importance resampling

Different particle filters are obtained if a different importance sampling distribution is chosen. Here we present the case

$$q(z_j | z_{j-1}^{(n)}, y_{0:j}) = p(z_j | z_{j-1}^{(n)}, y_j), \quad (3.6)$$

which results in the **optimal sequential importance resampling**. Its name is due to the fact that the proposal (3.6) minimises the variance of the ensemble weights $w_j^{(n)}$ conditional upon sample $z_{j-1}^{(n)}$ and the data $y_{0:j}$ (see [59, Proposition 2] for further details). With such optimal definition of importance sampling distribution, the importance weights can be updated according to

$$\hat{w}_j^{(n)} = w_{j-1}^{(n)} p(y_j | z_{j-1}^{(n)}). \quad (3.7)$$

The complete OPT-SIRS algorithm in case of nonlinear Gaussian signal model and Gaussian linear observation model is given by the following.

Initialise Initialise the OPT-SIRS filtering distribution, i.e. for $n = 1, \dots, N$, draw $z_0^{(n)} \sim \mathcal{N}(\mu_0, C_0)$ and set $\hat{p}_N(z_0) = \sum_{n=1}^N \frac{1}{N} \delta_{z_0^{(n)}}(z_0)$.

Then, for $j = 1, \dots, J$ apply:

Algorithm 3.6 Optimal importance resampling (OPT-SIRS)

```

1: function OPT-SIRS( $N, [\mu_0, C_0], [g_{0:J}(\cdot), \Sigma_z], [H, \Gamma], y_{0:J}$ )
2:   draw  $z_0^{(n)} \sim \mathcal{N}(\mu_0, C_0), \quad \forall n = 1 : N$  # Initialization
3:   for  $j = 1 : J$  do
4:      $\hat{\Sigma}_z \leftarrow (\Sigma_z^{-1} + H^T \Gamma^{-1} H)^{-1}$ 
5:      $\hat{\mu}^{(n)} \leftarrow \hat{\Sigma}_z (H^T \Gamma^{-1} y_j + \Sigma_z^{-1} g_{j-1}(z_{j-1}^{(n)})), \quad \forall n$ 
6:     draw  $\hat{z}_j^{(n)} \sim \mathcal{N}(\hat{\mu}^{(n)}, \hat{\Sigma}_z), \quad \forall n = 1 : N$ 
7:      $\hat{w}_j^{(n)} \leftarrow \exp\left(-\frac{1}{2} |y_j - H g_{j-1}(z_{j-1}^{(n)})|_{\Gamma + H \Sigma_z H^T}^2\right), \forall n$ 
8:     normalise  $\{\hat{w}_j^{(n)}\}$ 
9:     draw  $z_j^{(n)} \sim \sum_{n=1}^N \hat{w}_j^{(n)} \delta_{\hat{z}_j^{(n)}}(z_j), \quad \forall n = 1 : N$ 
return  $\left\{ z_{0:J}^{(n)} \right\}_{n=1}^N$ 

```

Prediction Update the ensemble particles according to the optimal importance distribution (3.6). Since we assume the observation model is Gaussian and linear, for each $n = 1, \dots, N$ the OPT-SIRS proposal distribution turns out to be Gaussian with mean $\hat{\mu}^{(n)}$ and covariance matrix $\hat{\Sigma}_z$ which can be easily computed through matrix multiplications and inversions.

Analysis Update the ensemble weights according to (3.7). Note that the linearity of the observation model guarantees a closed form for the updated weights. Then, set the proposed filtering distribution at time j to be $\sum_{n=1}^N \hat{w}_j^{(n)} \delta_{\hat{z}_j^{(n)}}(\hat{z}_j)$.

Resample Resample the predicted filtering distribution at time j in order to have an empirical distribution with uniform weights

$$\hat{p}_N(z_j | y_{0:j}) = \frac{1}{N} \sum_{n=1}^N \delta_{\hat{z}_j^{(n)}}(z_j)$$

Consult Algorithm 3.6 for a more compact summary of the OPT-SIRS algorithm.

Remark 3.3. *It should be emphasised that, although from a theoretical point of view the OPT-SIRS works for the general state-space model, it is of practical use only in case of (nonlinear) Gaussian signal model with linear observation operator. In fact, the proposal density defined by (3.6) can be easily sampled only if the signal model is Gaussian and H is a linear operator. Indeed, these are the hypotheses guaranteeing the normality of $p(\cdot | z_{j-1}^{(n)}, y_j)$.* ♠

In the following section, we mention a last notable family of particle filters, but further instances of sequential Monte Carlo methods can be found in [58]. Alternatively, for more recent overviews of the last advances in sequential Monte Carlo filtering we refer to [36], [215], and references therein. Also, the webpage of Signal Processing and Communications Group at the University of Cambridge [4] collects recent contributions on the topic of particle filters.

3.2.3 Gaussian particle filters

Here, we very briefly report that the combination of a Gaussian-like filter and the sequential Monte Carlo methodology produces interesting novel instances of methods: the Gaussian particle filters (GPFs) [123].

Indeed, by choosing a proposal distribution based on some nonlinear Gaussian filter, one can obtain very good estimation results while preserving the good theoretical properties of sequential Monte Carlo methods. Examples include the **unscented particle filter** (UPF) [202] whose proposal distribution is based on the unscented Kalman filter (see Section 3.1.2), but proposal distributions built on the extended Kalman [201], or on the ensemble Kalman filter [180] have been introduced as well.

In the following concluding section, we first spend a few words by mentioning the well-posedness of both smoothing and filtering problems, and then state some consistency result for particle filters.

3.2.4 Well-posedness and consistency of particle filters

Both smoothing and filtering problems are well-posed in the Hadamard sense [105, Definition 4.1]. In particular, we already established the **existence** of the filtering problem solution. Indeed, the prediction-analysis procedure pictured in Figure 3.1 illustrates how to define the filtering distribution iteratively in time through the functional representation of the prediction step (3.1) and the analysis step (3.2). In addition, under hypothesis which are guaranteed if the observation operator H introduced in (1.6) is a $L^2(\mathcal{Z})$ function, **stability** (i.e. weak continuity of the posterior distribution with respect to small variations in the dataset $y_{0:J}$) can be proved. See [134, Theorem 2.15, Corollary 2.16, Section 2.5] in this respect.

However, Sequential Monte Carlo methods also guarantee that the approximation introduced by these approximated algorithms is **consistent** (see [171, Section 2.2] for a definition of consistency of numerical methods in a general context). Indeed, it can be proved that under suitable hypotheses particle filters retrieve the exact filtering distribution in the large ensemble limit.

For instance, a preliminary result with a relatively simple proof guarantees that, if the observation operator H is bounded (i.e. $\exists \kappa \in (0, 1)$ such that $\kappa < H(z) < \kappa^{-1}$ for all $z \in \mathcal{Z}$), then the bootstrap filter approximation tends to the exact filtering distribution for $N \rightarrow \infty$. For the sake of brevity, we here omit all details on the type of the convergence. However, we report that the convergence rate is $C(\kappa, J) = \sum_{j=1}^J (\frac{2}{\kappa^2})^j$ which on the one hand is remarkably independent on the signal model dimension d_z (i.e., particle filters can, in principle, “beat the curse of dimensionality” [174]), but on the other hand it explodes in both limits $\kappa \rightarrow 0$ (observation operator assuming values arbitrary close to zero) and $J \rightarrow \infty$ (large time window). This means that the ensemble size should be extremely large in most applications, resulting in an essentially impracticable condition for real problems.

Remark 3.4. *The proof of such result is essentially based on three lemmas:*

- i) the Monte Carlo sampling operator defined in (3.5) introduces a $\mathcal{O}(N^{-1/2})$ -error between p and \widehat{p}_N ;*
- ii) the transition kernel associated to the signal model (i.e. the functional operator corresponding to (3.1)) is Lipschitz continuous¹ with $L = 1$;*
- iii) the likelihood operator resulting from the analysis step (3.2) is Lipschitz continuous with Lipschitz constant $L = 2\kappa^{-1}$. ♠*

Refer to [134, Theorem 4.5] and Lemma 4.7, Lemma 4.8, and Lemma 4.9 therein for further details on the theorem statement (such as the specific convergence metric) and the exact proof.

Some stronger results guaranteeing convergence of the approximated filtering distribution, along with bounds on the mean square errors under similar hypotheses are proved in [45]. Note the latter results apply to the general particle filters (i.e., not only to BF as Theorem 4.5 in [134], but also to OPT-SIRS and GPFs). In addition, all above-mentioned references only consider an autonomous signal state space (i.e., a transition density $f_{j-1}(z_j|z_{j-1})$ which is independent of the time variable j), whereas in our applications the signal variable is always time-dependent (cfr. Part III). Refer to [50] for a consistency result in case of time-dependent signal and observation models.

¹A function $f : X \rightarrow Y$ between two metric spaces $(X, d^{(x)})$ and $(Y, d^{(y)})$ is **Lipschitz continuous** with **Lipschitz constant** L if and only if

$$x_1, x_2 \in X \implies d^{(y)}(f(x_1) - f(x_2)) \leq L \cdot d^{(x)}(x_1 - x_2).$$

However, let us stress once again that in most practical applications (and the ones we present in Part III make no exception) it is not possible to meet the request of an observation operator which is bounded from below. Such remark highlights the need for more application-oriented convergence results, whose hypotheses on the transition and observation distributions are more realistic.

Chapter 4

Practical issues arising in data assimilation

In this last chapter of Part I, we address some issues which arise when the DA algorithms illustrated in the previous chapters are applied to solve some real-world problems.

First, in Section 4.1 we introduce the possibility for a state-space model to be parameter-dependent. Then, some strategies which can be employed to estimate such time-independent parameters are mentioned in Section 4.1.1. In addition, Section 4.1.2 illustrate how the state-space model can be redefined in order to allow several static parameters to be i.i.d. samples of some hyperparameter-dependent random variable.

Then, since in many applications the prior signal model is often expressed in form of ordinary differential equations (ODEs), Section 4.2 shows how the continuous-time ODE model can be turned into a discrete-time stochastic signal model. In particular, a detailed application of such procedure to a single-neuron model is included in the concluding Example 4.3.

Section 4.3 is devoted to show how approximate Gaussian filters – i.e., filtering methods which strongly rely on the unboundedness and the Gaussianity of the state-space model – can be applied to a bounded state-space by taking advantage of a suitable change of variables. The distortion introduced on the filtering distributions by such change of variables are investigated as well. We conclude Part I by introducing two approaches to evaluate the performances of a DA algorithm in Section 4.4. Refer [98] for a (non-exhaustive) overview of some other relevant issues arising when applying DA methods to practical problems.

4.1 Handling parameters

In the whole discussion we made in the preceding chapters, we focused on estimating the posterior distribution of the signal z , given the observation y . The actual target distribution was either the smoothing distribution $p(z_{0:J}|y_{0:J})$ or the filtering distribution $(p(z_{0:j}|y_{0:j}))_{j=0}^J$ in case of off- or on-line setting, respectively.

However, an important application of data assimilation addresses the parameter assessment of the proposed model. In fact, in real life, there is often some degree of uncertainty about the mathematical model that should be chosen to represent a given physical dynamics. In our framework, a typical approach consists in selecting a family of models indexed by some parameter θ , and then try to infer a set of good parameter values using the information contained in the data.

4.1.1 Data assimilation for parameter estimation

In order to stress the distinction between the signal and the model parameter, in this section we denote x the “dynamical” signal variable, and θ the “static” model parameter. Now, suppose that the initial distribution $p(x_0)$, the transition distribution of the signal model (1.3) $p(x_j|x_{0:j-1})$, and the conditional density of the observation model (1.4) $p(y_j|x_{0:j}, y_{0:j-1})$ depend on some parameter vector $\theta \in \Theta \subset \mathbb{R}^{d_\theta}$ which is not known exactly (i.e., θ is a random variable). This means that all probability densities are actually θ -dependent, namely

$$\begin{aligned} p(x_0) &\longrightarrow p_\theta(x_0) \\ p(x_j|x_{j-1}) &\longrightarrow p_\theta(x_j|x_{j-1}) \\ p(y_j|x_j) &\longrightarrow p_\theta(y_j|x_j), \end{aligned}$$

where the subscript notation actually stands for a further conditioning on the knowledge of the parameter value θ . Namely, $p_\theta(x_0)$ stands for $p(x_0|\theta)$, $p_\theta(x_j|x_{0:j-1})$ stands for $p(x_j|x_{0:j-1}, \theta)$, etc. Moreover, keep in mind that in the parameter-dependent state-space model analogous to model (1.3)-(1.4), the transition density for the non-homogeneous signal process not only depends on the discrete time j , but it also depends on the d_θ -dimensional parameter θ , i.e. $f_{j-1}(x_j|x_{j-1}) \longrightarrow f_{\theta_{j-1}}(x_j|x_{j-1})$.

Bayesian approach	Maximum likelihood approach
Off-line smoothing methods	
Particle marginal MH Augmented state space	Expectation maximization Gradient approach Likelihood function evaluation Iterated filtering
On-line filtering methods	
Practical filtering Augmented state space MCMC + SMC SMC ²	Expectation maximization Gradient approach

Table 4.1: Summary of DA methods for parameter estimation mentioned in [111]. Legend: MH=Metropolis Hastings, MCMC = Markov chain Monte Carlo, SMC = sequential Monte Carlo.

Example 4.1 (Parameter-dependent state-space model).

Suppose that the mean μ_0 and the covariance matrix C_0 of the stochastic initial condition in the Gaussian linear model defined in Example 1.2 are unknown. Then, the initial condition distribution is parameter-dependent, with parameter $\theta = (\mu_0, C_0)$. Indeed, the initial condition is $p_\theta(x_0) = f_\theta(x_0)$, where $f_\theta = \mathcal{N}(\theta)$. ♣

In such situation of parameter-dependent signal model, it is meaningful to estimate not only the dynamical signal $x_{0:j}$ underlying the observations $y_{0:j}$, but also the parameter value θ . There exist many different approaches which embed such new task to the signal estimation problem described in the previous chapters. The one we adopt in the experiments presented in Part III is the so-called augmented state-space model approach (see below). However, to give a hint of possible alternative strategies, in what follows we give a brief overview of the other possible approaches addressing the task of parameter estimation.

Following the two reviews [111, 110], we broadly classify DA methods for parameter estimation in Bayesian methods (BMs) and maximum likelihood methods. In addition, we also consider the distinction that we made in the previous chapters, i.e. between off-line and on-line algorithms.

In **Bayesian methods**, the parameter θ is considered as a random variable and, as such, it is endowed of a prior distribution $p(\theta)$. Then, a given BM is applied in order to approximate $p(\theta|y_{0:j})$, typically consisting in some kind of MCMC or SMC step. On the other hand, **maximum likelihood methods** provide a non-Bayesian estimator $\hat{\theta}$ that is given by the solution of

an optimisation problem. In the off-line case, such problem is given by

$$\hat{\theta} = \underset{\theta \in \Theta}{\operatorname{argmax}} \mathfrak{L}(\theta), \quad (4.1)$$

where $\mathfrak{L}(\theta) := \log(p_\theta(y_{0:J}))$, and we recall that $p_\theta(y_{0:J}) = p(y_{0:J}|\theta)$.

Now, we schematically describe the methods listed in [111, 110], while we refer to the original papers and references therein for further details on specific methods. Note that a comprehensive overview of such approaches is presented in Table 4.1.

Bayesian approach

Particle MCMC (*off-line*) In general, there are several MCMC methods relying on sequential Monte Carlo methods to build efficient proposal distribution. In particular, the reviews focus on a **particle marginal Metropolis-Hastings** (PMMH) which has a proposal density of the form $q((\hat{x}_{0:J}, \hat{\theta})|(x_{0:J}, \theta)) = q(\hat{\theta}|\theta)p_{\hat{\theta}}(\hat{x}_{0:J}|y_{0:J})$. Since it is impossible to sample directly such distribution¹, and it also yields an acceptance probability that is not available analytically, SMC methods are used at each iteration of the MCMC method in order to approximate the unknown terms.

Augmented state-space model (*off-/on-line*) In the **augmented state-space model** approach [116], the signal variable Z is augmented in order to include the new components corresponding to the parameter θ . In practice, one takes the signal variable to be the **augmented signal variable** $Z_j = (X_j, \theta_j)^T \in \mathcal{Z}$, where $\mathcal{Z} = \mathcal{X} \times \Theta \subset \mathbb{R}^{d_x} \times \mathbb{R}^{d_\theta}$. The signal model for the dynamical signal variable X remains the same as in (1.3), whereas the static θ -component is endowed of an artificial dynamics. In particular, the dynamics for θ_j is typically a random walk, e.g.

$$\theta_{j+1} \sim \mathcal{N}(\theta_j, \Sigma_{\theta_j}). \quad (4.2)$$

This means that $\theta_{j+1} = \theta_j + \epsilon_{\theta_j}$ where the noise random variable ϵ_{θ_j} in general has a time-dependent covariance matrix Σ_{θ_j} whose size (e.g. the largest eigenvalue) often decreases as a function of j .

Practical filtering (*on-line*) These are methods which rely on the fixed-lag approximation $p(x_{0:j-L}, \theta|y_{0:j-1}) \approx p(x_{0:j-L}, \theta|y_{0:j})$ and then run several MCMC in parallel in order to sample from a fixed-lag approximated filtering distribution.

¹In fact, $p_{\hat{\theta}}(\hat{x}_{0:J}|y_{0:J})$ is unknown

Using MCMC steps within SMC algorithms (*on-line*) Here, the introduction of the artificial parameter dynamics is avoided by reintroducing particle diversity via MCMC methods (adding, for instance, a Gibbs-sampler step on the parameter component at each SMC iteration).

SMC² (*on-line*) Such method was proposed in [42] and it is based on the superposition of two sequential Monte Carlo methods (hence, the name SMC²). The first SMC relies on N_θ particles in the θ space, and then, for each θ -particle, another SMC is run in the x -space. Note that, according to the authors, this is a sequential method but not a truly on-line one.

Maximum likelihood

Likelihood function evaluation (*off-line*) In order to evaluate the log-likelihood function $\mathcal{L}(\theta)$, we need to approximate it using some SMC methods. Indeed, in the optimisation problem (4.1) some unknown terms appear in the log-likelihood term. However, a straightforward problem is that the SMC approximation of $p_\theta(y_{0:J})$ is unbiased, but the corresponding approximation of $\mathcal{L}(\theta) = \log(p_\theta(y_{0:J}))$ is biased indeed. Thus, likelihood-function-evaluation methods indicate a series of expedients which can be employed in order to correct the source of bias (an example being correcting the resampling step of the SMC sub-algorithm).

Gradient approach (*off-line*) Whenever information on the gradient of the likelihood function $\mathcal{L}(\theta)$ is available, iterative steepest ascent methods of the form

$$\theta_{k+1} = \theta_k + \delta_{k+1} \nabla_\theta \mathcal{L}(\theta)|_{\theta=\theta_k},$$

can be used to identify the maximum point of the likelihood function. Here, δ_{k+1} is the step-size sequence of the steepest ascent method.

Expectation-maximization (*off-line*) The expectation-maximization (EM) framework consists in an iterative maximization problem which is articulated in two steps. First, the objective cost function

$$\mathcal{Q}(\theta_k, \theta) = \int_{\mathcal{X}^{J+1}} \log(p_\theta(x_{0:J}, y_{0:J})) p_{\theta_k}(x_{0:J}|y_{0:J}) dx_{0:J}$$

is computed, usually resorting to some trick in order to approximate the integral. Note that this is called the expectation step (E-step) because the integral can be interpreted as the expectation of the function $\log(p_\theta(x_{0:J}, y_{0:J}))$ with respect to the probability distribution $p_{\theta_k}(x_{0:J}|y_{0:J}) dx_{0:J}$.

Then, the optimisation update is computed in the maximization step (M-step) by solving the maximization problem $\theta_{k+1} = \operatorname{argmax}_{\theta \in \Theta} \mathcal{Q}(\theta_k, \theta)$.

Iterated filtering (*off-line*) The iterated filtering approach, relies on an approximation of $\nabla_{\theta} \mathcal{L}(\theta)|_{\theta=\theta_j}$, the gradient likelihood evaluated in θ_j , the approximated parameter value at iteration j . Such approximation is based on the posterior moments of an artificial augmented state-space model $Z_j = (X_j, \tilde{\theta}_j)$. Such approximation is then used to apply a gradient ascent method, even when information on the gradient of the likelihood function is not available.

Gradient approach (*on-line*) As in the off-line case, if the gradient of the likelihood is available, such information can be used to implement a steepest ascent algorithm. However, in the on-line case we consider a sequential version of the log-likelihood, namely

$$\theta_{j+1} = \theta_j + \delta_{j+1} \nabla_{\theta} \log \left(p_{\theta_{0:j}}(y_j | y_{0:j-1}) \right).$$

Expectation-maximization (*on-line*) In case $p_{\theta}(x_{0:j}, y_{0:j})$ is in the exponential family², one can implement an expectation maximization algorithm similar to the one described in the off-line case. However, the definition of the function \mathcal{Q} should be adjusted to reflect the on-line methodology.

We now concluded our overview of available approaches to deal with parameter-dependent transition densities. In the following section we introduce the concept of hyperparameter within the context of data assimilation.

4.1.2 State-space model for hyperparameters estimation

In Chapter 1, it was showed that a general state-space model can be schematically represented as the directed acyclic graph pictured in Figure 1.1. When

² A parameter-dependent state-space model is in the exponential family if, for all $x_0, \tilde{x}_0 \in \mathcal{X}$, for all j and all observations $y_{0:j} \in \mathcal{Y}^{j+1}$, there exist $C > 0$ and $\lambda \in [0, 1)$ such that

$$\int_{\mathcal{X}} \left| p_{\theta}(x_j | y_{0:j}, x_0) - p_{\theta}(x_j | y_{0:j}, \tilde{x}_0) \right| dx_j \leq C \lambda^j.$$

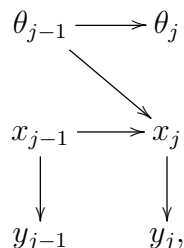
focusing on parameter estimation and adopting an augmented state-space approach as described in the preceding section at page 60, the signal model (1.1) writes in the following way. First, the initialization step becomes

$$\begin{cases} \theta_0 \sim p(\theta_0) \\ X_0 \sim p(x_0 | \theta_0) \end{cases},$$

Then, the j -th update step can be easily adapted to

$$\begin{cases} \theta_j \sim p(\theta_j | \theta_{j-1}) \\ X_j \sim p(x_j | x_{j-1}, \theta_{j-1}) \end{cases}, \quad j > 0.$$

This produces the following DAG



which corresponds to a state-space model where the Markov property is preserved. Indeed, the conditional distribution of the augmented state variable $Z_j = (X_j, \theta_j)^T$ still only depends on (X_{j-1}, θ_{j-1}) , the state of the system at time $j - 1$.

We now ask ourselves, how does the DAG changes if we consider a situation where multiple parameter components $\theta^{(1)}, \theta^{(2)}, \dots$ are actually i.i.d. samples of the same λ -indexed distribution $p_\lambda(\theta^{(i)}) = p(\theta^{(i)} | \lambda)$? Note that such a λ is called **hyperparameter**, as it is the parameter characterising the distribution of a whole collection of model parameters $\theta^{(i)}$ for $i = 1, 2, \dots$. In what follows, we show that considering hyperparameters introduces a further level in the DAG representation, and that we need to specify both parameter and hyperparameters dynamics for such a SSM. In addition, let us anticipate that the corresponding dynamics on the parameters should preserve the parameter/hyperparameter relationship in some sense.

For the sake of readability, we present the framework which allows one to define an augmented state-space model only in case of one single hyperparameter $\lambda \in \Lambda \subset \mathbb{R}$ which characterises the distribution of d_θ parameters $\theta^{(1)}, \dots, \theta^{(d_\theta)}$. The case of multiple hyperparameters, each one with a different number of corresponding parameters is trivial to generalise intuitively, but introducing a consistent notation entails a significantly cumbersome adaptation.

Moreover, in the current section the probability density function for the i.i.d. sequence $(\theta^{(i)})_{i=1}^{d_\theta}$ is denoted $\varphi(\theta | \lambda)$, whereas we write $F(\theta | \lambda)$ to indicate the cumulative distribution function (c.d.f.). Namely,

$$\begin{aligned} P\left[\theta^{(i)} \in [\theta, \theta + d\theta] \mid \lambda\right] &= \varphi(\theta | \lambda) d\theta, \\ P\left[\theta^{(i)} \leq \theta \mid \lambda\right] &= F(\theta | \lambda), \end{aligned}$$

for all $i = 1, \dots, d_\theta$. The inverse of the cumulative distribution function (also known as **quantile function**) is denoted $F^{-1}(q | \lambda)$, for $q \in [0, 1]$.

In this work, we propose an arbitrary initialization for the hyperparameter and the induced random initialization for the corresponding parameters, i.e.

$$\begin{aligned} \lambda_0 &\sim p(\lambda_0) \\ \theta_0^{(i)} &\sim \varphi(\theta | \lambda_0), \quad i = 1, \dots, d_\theta. \end{aligned}$$

As for the dynamics iteration, we consider a stochastic Gaussian random-walk dynamics for the hyperparameter λ , i.e. for $j > 0$

$$\lambda_j \sim \mathcal{N}(\lambda_{j-1}, \Sigma_{\lambda_{j-1}}).$$

However, the induced parameter dynamics is not random at all. On the contrary, it can univocally determined if the previous parameter value and both current and previous hyperparameter values are given. Indeed, by simply applying the λ_{j-1} -c.d.f. and the inverse λ_j -c.d.f. the following update rule is obtained

$$\theta_j^{(i)} = F^{-1}\left(F(\theta_{j-1}^{(i)} | \lambda_{j-1}) \mid \lambda_j\right).$$

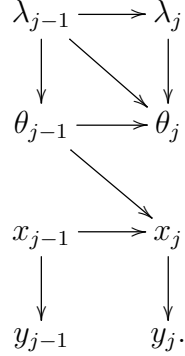
Using the shorthand notation $F_\lambda(\theta) = F(\theta | \lambda)$ and $F_\lambda^{-1}(q) = F^{-1}(q | \lambda)$, the above update rule can be written in the concise form

$$\theta_j^{(i)} = F_{\lambda_j}^{-1} \circ F_{\lambda_{j-1}}(\theta_{j-1}^{(i)}). \quad (4.3)$$

Such update rules consists in projecting the previous parameter values in the quantile space according to the previous hyperparameter value (i.e. $F_{\lambda_{j-1}}(\theta_{j-1}^{(i)})$) and then converting such quantile into a new parameter value according to the new hyperparameter value (the $F_{\lambda_j}^{-1}$ -term).

As a result, the DAG for the augmented state-space model with hyperpa-

rameters and parameter dynamics given by (4.3) is



We observe that such DAG structure apparently breaks the Markovian assumption for the augmented state variable $Z = (X, \theta, \lambda)$. In fact, the conditional distribution of θ_j depends on λ_j and not only on λ_{j-1} and θ_{j-1} . However, since the value of θ_j is not needed until the following $x_j \rightarrow x_{j+1}$ update, we can bypass the problem (even though this is only from an implementation point of view) by computing such value only when it is actually needed.

Also, we note that if we record the quantiles corresponding to the initial parameter realizations, the update rule is only dependent on such values and the new hyperparameter value. In fact, conditional on the event

$$\left\{ \theta_0^{(i)} = \check{\theta}_0^{(i)} : i = 1, \dots, d_\theta \right\},$$

if we set $\check{q}_0^{(i)} = F_{\lambda_0}(\check{\theta}_0^{(i)})$ for $i = 1, \dots, d_\theta$, then for all $j > 0$

$$\theta_j^{(i)} = F_{\lambda_j}^{-1}(\check{q}_0^{(i)}).$$

This can be easily proved by an induction argument.

Let us conclude the current section with an example.

Example 4.2 (LIF network).

Consider the LIF network model (6.4) defined and detailed in Part II as a demonstrative example. Focusing on the leakage constants of excitatory cells, we assumed that they follow an exponential distribution of parameter L_E^\dagger , i.e. $L_1, \dots, L_{n_E} \sim \text{Exp}(L_E^\dagger)$. Hence, in this case we have that the hyperparameter is $\lambda = L_E^\dagger$, the corresponding parameters $\theta^{(i)} = L_{E,i}$, $d_\theta = n_E$, the probability density function is given by

$$\varphi(L_E | L_E^\dagger) = \begin{cases} \frac{1}{L_E^\dagger} e^{-L_E/L_E^\dagger} & L_E \geq 0 \\ 0 & L_E < 0 \end{cases},$$

the cumulative distribution function by

$$F(L_E | L_E^\dagger) = \begin{cases} 1 - e^{-L_E/L_E^\dagger} & L_E \geq 0 \\ 0 & L_E < 0 \end{cases},$$

and the inverse cumulative distribution function, defined for $q \in [0, 1)$, by

$$F^{-1}(q | L_E^\dagger) = -L_E^\dagger \ln(1 - q)$$



We now move to showing how to build a discrete-time state-space model starting from a parameter-dependent ODE model.

4.2 Twin experiments from continuous-time models

In this section we describe how to retrieve a discrete-time Gaussian nonlinear state-space model of the form (1.5) from a continuous-time ODE model. We do so in the situation where we are interested in the estimating the parameters of the ODE model, and where we adopt an augmented state-space model approach as described in Section 4.1.1 at page 60.

In what follows, we first present the procedure which can be applied to a general ODE model. Then, in Example 4.3, we instantiate such procedure by providing the discrete-time random state-space model for a single neuron model.

ODE model declaration First of all, we write the ODE model we wish to “discretise” and “randomise” in the form

$$\dot{x} = F(t, x; \theta), \quad t \in [0, T_f], \quad (4.4)$$

where F is the non-autonomous θ -dependent vector field of the ODE model (4.4), t is the independent time variable, x is the vector of the state variables, and θ is the vector of the modelling parameters. Note that the ODE model is supposed to be coupled to a data model $y(t) = H_x \circ x(t)$ for some operator H_x .

Augmentation Then, we put parameter vector θ into the same footing as a state variable according to the augmented state-space approach described at page 60. This means that we first augment system (4.4) with the extra dynamic

equation $\dot{\theta} = 0$, so that $\theta = \theta(t)$. As a consequence, the ODE equation (4.4) becomes the augmented ODE system

$$\begin{cases} \dot{x} = F(t, x; \theta) \\ \dot{\theta} = 0 \end{cases}.$$

Note that the new augmented state variable is $z(t) = (x(t), \theta(t))^T$.

Discretization Now, we fix a numerical step Δt , a time mesh $t_{0:J} = (t_j)_{j=0}^J$ such that $t_j = j\Delta t$ and $t_J = T_f$. Also, we choose a numerical approximation scheme for ODEs systems (such as the forward or backward Euler method or a more accurate Runge-Kutta method). This allows us to map the continuous-time solution $(x(t), \theta(t))_{t \in [0, T_f]}$ into the discrete-time approximation $(\tilde{x}_j, \tilde{\theta}_j)_{j=0}^J$, where \tilde{x}_j and $\tilde{\theta}_j$ are the numerical approximation of $x(t_j)$ and $\theta(t_j)$, respectively.

Then, the augmented ODE system becomes the (deterministic) discrete-time dynamical system

$$\begin{cases} \tilde{x}_j = g_{j-1}(\tilde{x}_{j-1}; \tilde{\theta}_{j-1}) \\ \tilde{\theta}_j = \tilde{\theta}_{j-1} \end{cases}, \quad j = 1, \dots, J;$$

where $g_{j-1}(\tilde{x}_{j-1}; \tilde{\theta}_{j-1})$ is the discrete-time version of the continuous-time vector field $F(t, x; \theta)$. Note that j is the discrete-time variable corresponding to the continuous-time variable t . Moreover, let us stress the fact that the exact analytical form of map g_{j-1} depends on

- (i) the original vector field F ;
- (ii) the numerical method chosen.

Loosely speaking, we occasionally refer to either the discrete map g_{j-1} or the original continuous-time vector field F as the **prior model** in what follows. The discretised augmented state variable is now $z_j = (x_j, \theta_j)^T$.

Randomization Finally, we turn both parameter and state variable into random processes by considering the presence of an additive random noise in the model dynamics.

Namely, if the signal space is unbounded (i.e. $\mathcal{Z} = \mathbb{R}^{d_z}$, where $\mathcal{Z} = \mathcal{X} \times \Theta \subset \mathbb{R}^{d_x} \times \mathbb{R}^{d_\theta}$), the parameter-dependent transition rule for the discrete-time stochastic process $(X_j, \theta_j)_{j \geq 0}$ writes

$$X_j \mid \left(X_{0:j-1} = \tilde{x}_{0:j-1}, \theta_{0:j-1} = \tilde{\theta}_{0:j-1} \right) = g_{j-1}(\tilde{x}_{j-1}; \tilde{\theta}_{j-1}) + \epsilon_{x_j}, \quad (4.5)$$

where $(\epsilon_{x_j})_{j>0}$ is an i.i.d. sequence independent on X_0 , with $\epsilon_{x_j} \sim \mathcal{N}(0, \Sigma_x)$. In the same way, the signal model for the parameter component θ is a Gaussian random walk, with transition equation

$$\theta_j \mid \left(X_{0:j-1} = \tilde{x}_{0:j-1}, \theta_{0:j-1} = \tilde{\theta}_{0:j-1} \right) = \tilde{\theta}_{j-1} + \epsilon_{\theta_j},$$

where $(\epsilon_{\theta_j})_{j>0}$ is an i.i.d. sequence independent on X_0 , on θ_0 and on $(\epsilon_{x_j})_{j>0}$, with $\epsilon_{\theta_j} \sim \mathcal{N}(0, \Sigma_\theta)$.

At the end of such randomization procedure, the stochastic discretised augmented state variable is $Z_j = (X_j, \theta_j)^T$, whereas the update rule (4.5) is Gaussian and, in general, nonlinear.

In conclusion, we can as well introduce the observation model

$$Y_j \mid \left(X_{0:j} = \tilde{x}_{0:j}, \theta_{0:j} = \tilde{\theta}_{0:j}, Y_{0:j} = y_{0:j} \right) = H_x(\tilde{x}_j) + \varepsilon_j,$$

which is defined over the observation space $\mathcal{Y} = \mathbb{R}^{d_y}$. In addition, in the above equation $H_x : \mathcal{X} \rightarrow \mathcal{Y}$ denotes the original nonlinear observation operator, $(\varepsilon_j)_{j>0}$ is an i.i.d. sequence independent on $(\epsilon_{x_j})_{j>0}$, on X_0 , and on θ_0 , with $\varepsilon_j \sim \mathcal{N}(0, \Gamma)$.

Let us now illustrate an example of such procedure on a given ODE system.

Example 4.3 (Discrete SSM for the single-neuron model (5.8)).

As an example, keep in mind the single-neuron toy model

$$\begin{cases} \dot{V} &= \left[-\bar{g}_K a(V - E_K) - \bar{g}_{Na} b_\infty(V)(V - E_{Na}) \right. \\ & \left. - \bar{g}_L(V - E_L) + I_{\text{ext}}(t) \right] / C, \\ \dot{a} &= [a_\infty(V) - a] / \tau_a, \end{cases} \quad \forall t \in [0, T_f]. \quad (4.6)$$

Details about such model, its state variables, its modelling parameters and what they represent are given in Section 5.2. At this point, it suffice to say that the continuous-time state variable is $x = (V, a)^T$, whereas the vector of parameters we wish to estimate is given by the following list of ionic parameters $\theta = (\bar{g}_{Na}, E_{Na}, \bar{g}_K, E_K, \bar{g}_L, E_L, K^{(b)}, V_{1/2}^{(b)}, K^{(a)}, V_{1/2}^{(a)})^T$. Note that the scale parameters C and τ_a are considered known and fixed.

(Augmentation and discretization) Then, we choose the fourth order Runge-Kutta solver for ODEs over the time mesh $t_{0:J} = (t_j)_{j=0}^J$, where $t_j = j\Delta t$, $\Delta t = 0.01$ ms, and $T_f = t_J = 500$ ms. As a consequence, the exact analytical form of the (deterministic) augmented discretised version of (4.6) is

$$\begin{cases} \tilde{x}_j = \tilde{x}_{j-1} + \frac{\Delta t}{6} [k_1 + 2k_2 + 2k_3 + k_4] =: g_{j-1}(\tilde{x}_{j-1}, \tilde{\theta}_{j-1}) \\ \tilde{\theta}_j = \tilde{\theta}_{j-1} \end{cases}$$

where

$$\begin{aligned}
k_1 &= F(t_{j-1}, \tilde{x}_{j-1}; \tilde{\theta}_{j-1}); \\
k_2 &= F\left(t_{j-1} + \frac{\Delta t}{2}, \tilde{x}_{j-1} + \frac{\Delta t}{2}k_1; \tilde{\theta}_{j-1}\right); \\
k_3 &= F\left(t_{j-1} + \frac{\Delta t}{2}, \tilde{x}_{j-1} + \frac{\Delta t}{2}k_2; \tilde{\theta}_{j-1}\right); \\
k_4 &= F(t_{j-1} + \Delta t, \tilde{x}_{j-1} + \Delta tk_3; \tilde{\theta}_{j-1});
\end{aligned}$$

and

$$F : (0, +\infty) \times \mathcal{X} \times \Theta \longrightarrow \mathcal{X}$$

$$(t, x; \theta) \longrightarrow F(t, x; \theta)$$

denotes the r.h.s. of model (4.6) written in vectorial form.

(Randomization) Finally, randomization implies the following model dynamics for the stochastic vector $z_j = (x_j, \theta_j)^T$

$$\begin{cases} x_j = g_{j-1}(x_{j-1}, \theta_{j-1}) + \epsilon_{x_j} \\ \theta_j = \theta_{j-1} + \epsilon_{\theta_j}, \end{cases} \quad j \in \{1, \dots, J\}; \quad (4.7)$$

where we recall that

- x_j and θ_j are the discrete-time approximate (and noisy) values corresponding to $x(t_j)$ and $\theta(t_j)$ respectively;
- $g_{j-1} : \mathbb{R}^{d_x} \times \mathbb{R}^{d_\theta} \longrightarrow \mathbb{R}^{d_x}$ represents the discrete version of vector field (4.6) obtained by applying the fourth order Runge-Kutta method for ODEs. Because of the time-dependent term $I_{\text{ext}}(t)$ in the neuron model, the vector field is non-autonomous and g is indeed j -dependent.
- the dimension of variables component x_j is $d_x = 2$ and the dimension of parameter the component θ_j is $d_\theta = 10$;
- $\{\epsilon_{x_j}\}_{j=1}^J$ and $\{\epsilon_{\theta_j}\}_{j=1}^J$ are two mutually independent sequences of i.i.d. random variables with $\epsilon_{x_1} \sim \mathcal{N}(0, \Sigma_x)$ and $\epsilon_{\theta_1} \sim \mathcal{N}(0, \Sigma_\theta)$.

(Random initial condition) In conclusion, in order to match the general form of the nonlinear Gaussian signal model (1.5), the initial condition can be set to be a Gaussian random variable

$$\begin{cases} x_0 \sim \mathcal{N}(\mu_{x_0}, C_{x_0}) \\ \theta_0 \sim \mathcal{N}(\mu_{\theta_0}, C_{\theta_0}), \end{cases} \quad (4.8)$$

where μ_{x_0} and μ_{θ_0} are the variables and parameter component of the initial mean, respectively; whereas C_{x_0} and C_{θ_0} are the corresponding covariance matrices.

Moreover, we can complete the signal state-space model (4.7) with a set of noisy observations $\{y_j\}_{j=1}^J$ linked to the state variable through the data model

$$y_j = H_x x_j + H_\theta \theta_j + \varepsilon_j. \quad (4.9)$$

Here, $H_x : \mathbb{R}^{d_x} \rightarrow \mathbb{R}^{d_y}$ and $H_\theta : \mathbb{R}^{d_\theta} \rightarrow \mathbb{R}^{d_y}$ are linear operators (i.e. a $d_y \times d_x$ and $d_y \times d_\theta$ matrices, respectively) with $d_y = 1$, and $(\varepsilon_j)_{j=1}^J$ is an i.i.d. sequence with $\varepsilon_1 \sim \mathcal{N}(0, \Gamma)$. Since parameters cannot be directly observed, H_θ is set to be a $d_y \times d_\theta$ matrix with zero-entries. Also, as we exemplify in Section 5.2.1, a typical choice for H_x is the projection on the first component, which corresponds to the measurement of the membrane potential, i.e. $H_x = (1, 0)$. Note that what was obtained is a data model of the form (1.6), with a linear observation operator. ♣

We here conclude the description of how to convert a deterministic and continuous-time model into a discrete-time stochastic SSM.

In the following section we address a different but related matter. Indeed, we propose a naive methodology to transform a bounded state-space model into an unbounded one. The motivation is that, however simple, the machinery we introduce allows one to employ Gaussian DA methods also for models which include bounded variables or bounded parameters.

4.3 Dealing with bounded variables

As we exemplify in Part II and Part III, in many applications some of the variables of a model are physically constrained to lie in some interval. For instance, this is the case for all the conductance-based single-neuron models we present in Chapter 5. In fact, the activation variables m_{ion} 's need to lie in the interval $(0, 1)$ as they represent the opening probabilities of the corresponding ionic channels. In addition, another common case is when some variable is subject to a positivity constraint. As an example, consider the internal calcium concentration $[Ca^{2+}]_{\text{in}}$, which has a dynamics described by (5.5) and it is involved in the morphological model (5.11)-(5.14). As a consequence, the state space $\mathcal{Z} \subset \mathbb{R}^{d_z}$ for such neural models is a bounded subset of the real space. This poses the problem of designing signal processes which enforce such bonds within a stochastic framework.

In what follows we discuss some noise-design options in the case where the

signal process follows a Gaussian nonlinear model as in Example 1.3 so that

$$z_j = g_{j-1}(z_{j-1}) + \epsilon_j,$$

where $(\epsilon_j)_{j>0}$ is the i.i.d. dynamical noise process. Also, for the sake of simplicity, we assume that the signal state space is unidimensional ($d_z = 1$) and that the signal itself satisfies

- i)* a single inequality constraint $z > a$ so that $\mathcal{Z} = (a, \infty)$; or
- ii)* two inequalities $z > a$ and $z < b$, so that z lives in the bounded open interval $\mathcal{Z} = (a, b)$.

Of course, we assume that $a, b \in \mathbb{R}$ are finite real numbers with $a < b$. It is straightforward to generalise what we state to the multi-dimensional case where

$$\mathcal{Z} = \prod_{i=1}^{d_{z_1}} (a_i, b_i) \times \prod_{i=1}^{d_{z_2}} (a_i, \infty) \times \mathbb{R}^{d_{z_3}}$$

is the direct product of an d_{z_1} -dimensional hyper-cube, d_{z_2} half lines, and a d_{z_3} -dimensional hyper-spaces, with $d_{z_1} + d_{z_2} + d_{z_3} = d_z > 1$.

There are several approaches which can enforce boundedness of some specific variable. One option is to select a distribution for the dynamical noise ϵ_j that does not drive the model out of its physical bounds. For instance, in [56, 55] the authors propose a particle filter for a Morris-Lecar neuron model where the noise acting on the bounded activation variable has a standard deviation that approaches zero when the activation variable approaches the boundaries of the state space. Another possibility proposed in [20, 170] is to assume that the signal process follows a truncated normal distribution³ $p(z_j | z_{j-1}) = \mathcal{N}_{(a,b)}(g_{j-1}(z_{j-1}), \sigma_z^2)$, where b can be either finite or infinite.

However, the first solution is possible only due to the specific form of the activation variable dynamics and when considering a stochastic differential equation framework [54], which should be possible to apply to the models we

³A random variable X is said to follow a normal distribution with parameters μ and σ^2 truncated on the interval (a, b) , if its probability density function is $p(x) = \sigma^{-1} f(\frac{x-\mu}{\sigma}) [F(\frac{b-\mu}{\sigma}) - F(\frac{a-\mu}{\sigma})]^{-1}$ for $x \in (a, b)$ and zero otherwise, where $f(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$ is the p.d.f. of a standard unit normal r.v. and $F(q) = \int_{-\infty}^q f(x) dx$ is its cumulative density function. In such a case we write $X \sim \mathcal{N}_{(a,b)}(\mu, \sigma^2)$. These definitions required finite $a, b \in \mathbb{R}$, but it suffice to take the suitable limits of the c.d.f. and the definition is valid also for $a = -\infty$ or $b = +\infty$. Note that, loosely speaking, truncating only entails multiplying by the characteristic function $\mathbb{1}_{(a,b)}$ and renormalizing the resulting p.d.f.

For a more detailed introduction to truncated normal distributions we refer to [101]. See [27] for the discussion of some numerical issues related to the simulation of such distributions.

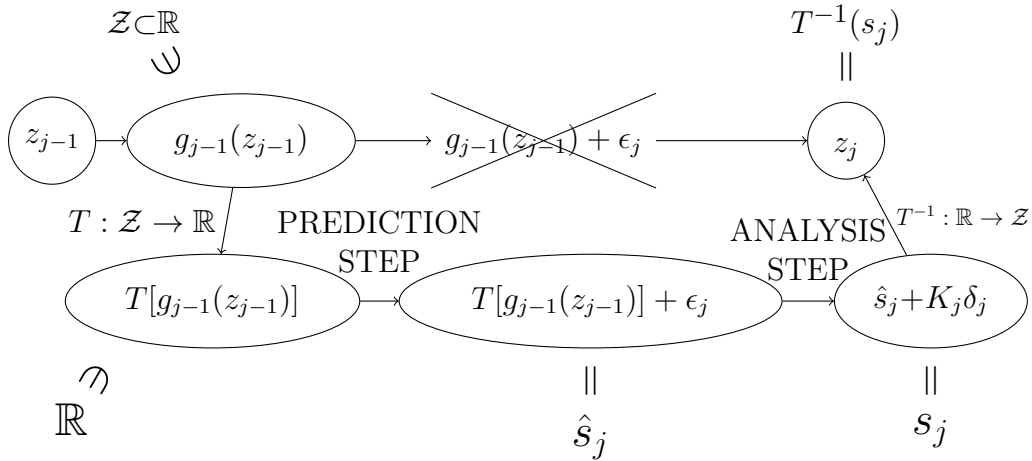


Figure 4.2: Graphical representation of the Kalman filter in case of bounded variables. Compare with the equivalent figure in case of unbounded variables Figure 3.3.

consider in theory but of cumbersome implementation when using the Neuron simulation environment (see Section 5.1). On the other hand, the truncated-Gaussian approach is somewhat arbitrarily artificial, and should break, in principle, the theoretical results available for Gaussian and approximate Gaussian methods.

As a consequence, we propose a different approach which is just as much artificial, but at least preserves the Gaussian structure of the state-space model in its integrity. The key idea is very simple and consists in mapping the bounded state space $\mathcal{Z} \subset \mathbb{R}$ onto the whole real line through a suitable invertible transformation $T : \mathcal{Z} \rightarrow \mathbb{R}$ before considering the dynamical noise. Then, applying the prediction step on the transformed variable s_j (instead of the original variable z_j) and return to the “physical” variable $z_j = T^{-1}(s_j)$ only after applying the analysis step guarantees that original variable z never exit its physical bounds. Such procedure is schematically represented in Figure 4.2.

Note that this approach is equivalent to substituting the original bounded variable z with its real-valued counterpart s (by applying to the change of variable $s = T(z)$), and then adjusting the time map accordingly, i.e. taking $\tilde{g}_{j-1}(s) = g_{j-1} \circ T^{-1}(s)$. However, from the point of view of the original physical variable z_j , applying the noise $\epsilon_j \sim \mathcal{N}(0, \sigma^2)$ on its real-valued counterpart s_j modifies the state-space model and introduces some distortion in the noise acting on the physical variable.

We now investigate what noise distortions are introduced in two examples: in the first one we can compute analytically the moments and the distribution of the noise acting on the physical variable z ; in the second example, where

\mathcal{Z}	$T(z)$	$T^{-1}(s)$	$\frac{p(z_j z_{j-1})}{\mathbb{E}[Z_j z_{j-1}]}$	$\frac{\log -\mathcal{N}(\log(\mu), \sigma^2)}{\mu \exp(\sigma^2/2)}$
$(0, \infty)$	$\log(z)$	$\exp(s)$	$\text{Var}[Z_j z_{j-1}]$	$\mu^2(e^{\sigma^2} - 1)e^{\sigma^2}$
(a, ∞)	$\log(z-a)$	$a + e^s$	$\text{Skew}[Z_j z_{j-1}]$	$\mu^{3/2}(e^{\sigma^2} + 2)\sqrt{e^{\sigma^2} - 1}$
$(0, 1)$	$\text{logit}(z)$	$\ell(s)$	$\text{Kurt}[Z_j z_{j-1}]$	$e^{4\sigma^2} + 2e^{3\sigma^2} + 3e^{2\sigma^2} - 3$
(a, b)	$\log\left(\frac{z-a}{b-z}\right)$	$\frac{a+be^s}{1+e^s}$		

(a) Transformations considered in this chapter to deal with bounded variables. Note that, $\text{logit}(z) = \log\left(\frac{z}{1-z}\right)$ and $\ell(s) = \frac{1}{1+e^{-s}}$.

(b) Moments of a log-normal r.v. Note that $\mu = g_{j-1}(z_{j-1})$, the deterministically predicted state at time j , enters the transition distribution as a scaling factor. Consequently, the mean scales linearly with μ and the variance scales quadratically.

Table 4.3: Change of variables for bounded state spaces and moments of a log-normal random variable.

analytical computation are not possible, we illustrate the dependence of various moments of the noise distribution via numerical integration.

4.3.1 Single inequality constraint

Let us start off by the simple case of a single unidimensional positive variable $z \in \mathcal{Z} = (0, +\infty)$. A possible choice is to resort to the logarithmic function $T(z) = \log(z)$ with inverse $T^{-1}(s) = \exp(s)$. Because of the group-homomorphism property of $T(z) = \log(z)$, this translates in a log-normal noise acting on the physical variable z_j . Indeed,

$$\begin{aligned}
z_j &= T^{-1}\left[T[g_{j-1}(z_{j-1})] + \epsilon_j\right] \\
&= \exp\left(\log(g_{j-1}(z_{j-1})) + \epsilon_j\right) \\
&= g_{j-1}(z_{j-1}) \cdot \exp(\epsilon_j).
\end{aligned} \tag{4.10}$$

We observe that, as in the regular nonlinear Gaussian state-space model, the signal model can be divided in the deterministic part $g_{j-1}(z_{j-1})$ and a stochastic multiplicative noise $\epsilon'_j = \exp(\epsilon_j)$.

The log-normal is a well-known distribution, and its moments⁴ can be computed in closed form (see Table 4.3b). In fact, we know for instance that the

⁴A r.v. X with expected value $\mathbb{E}[X] = \mu$ and variance $\text{Var}[X] = \sigma^2$ has skewness defined by $\text{Skew}[X] = \mathbb{E}\left[\left(\frac{X-\mu}{\sigma}\right)^3\right] = \frac{\mathbb{E}[(X-\mu)^3]}{\sigma^3}$. The skewness represents the degree of asymmetry of a distribution: a negative (positive) skew indicates that the distribution has a right-leaning p.d.f (resp. left-leaning) i.e. the distribution mass is concentrated on the right (resp. left). The kurtosis is defined as $\text{Kurt}[X] = \frac{\mathbb{E}[(X-\mu)^4]}{\text{Var}[X]^2}$ and it measures the size of the distribution

multiplicative log-normal noise ϵ'_j has mean $\mathbb{E}[\epsilon'_j] = e^{\sigma^2/2}$ which is greater than one for $\sigma > 0$. This means that the physical variable will tend to have slightly larger values than the ones the deterministic model $z_{j-1} \mapsto g_{j-1}(z_{j-1})$ would predict.

The more general case where z has to satisfy the bound $z > a$ (with $a \in \mathbb{R}$) can be dealt with in a completely analogous way by considering the shifted transformation $T(z) = \log(z - a)$.

4.3.2 Bounded-interval constraint

Let us move to the case of signal variable which is bounded from both below and above, starting off by the case where $(a, b) = (0, 1)$. A suitable change of variable mapping $(0, 1)$ onto \mathbb{R} is the logit transformation defined by

$$\text{logit}(z) = \log\left(\frac{z}{1-z}\right).$$

Note that this is the overlap of two logarithms: one centred on $z = 0$ and one centred on $z = 1$ with inverted sign ($+\log(z)$ and $-\log(1-z)$, respectively). The inverse of the logit function is given by the S-shaped (i.e. sigmoid) logistic function

$$\ell(s) = \frac{\exp(s)}{1 + \exp(s)}.$$

For a graphical inspection of such functions, refer to Figure 4.4. The left panel pictures the graph of the logit function, whereas the sigmoid function $\ell(s)$ is represented in the right panel.

Now, let us denote $\mu = g_{j-1}(z_{j-1})$ and $S_j = \text{logit}(\mu) + \epsilon_j$. Then, the r.v. $Z_j = \ell(S_j)$ follows a logit-normal distribution⁵ with location parameter $\text{logit}(\mu)$ and scale parameter σ^2 . Unfortunately, in this case no group-homomorphism property is available, and the deterministic and stochastic parts of the signal model are hopelessly entangled. What is more, no closed form for the moments of the logit-normal distribution exists.

In order to visualise the shape of the logit-normal distribution, we compute numerically its probability density function for different values of the untransformed location parameter μ (and fixed scale $\sigma^2 = 0.01$). The results illustrated in Figure 4.5a show that the location parameter has a strong influence on the shape of the p.d.f. In particular, the most conspicuous effect is

tails: a kurtosis smaller (greater) than three denotes tails that are thinner (resp. fatter) than those of a Gaussian r.v., resulting in less extreme (resp. more extreme) and less frequent (resp. more frequent) outliers.

⁵In fact, a random variable X is said to be logit-normal distributed if $\text{logit}(X) \sim \mathcal{N}(m, \sigma^2)$. In this case we write $X \sim \text{logit-}\mathcal{N}(m, \sigma^2)$.

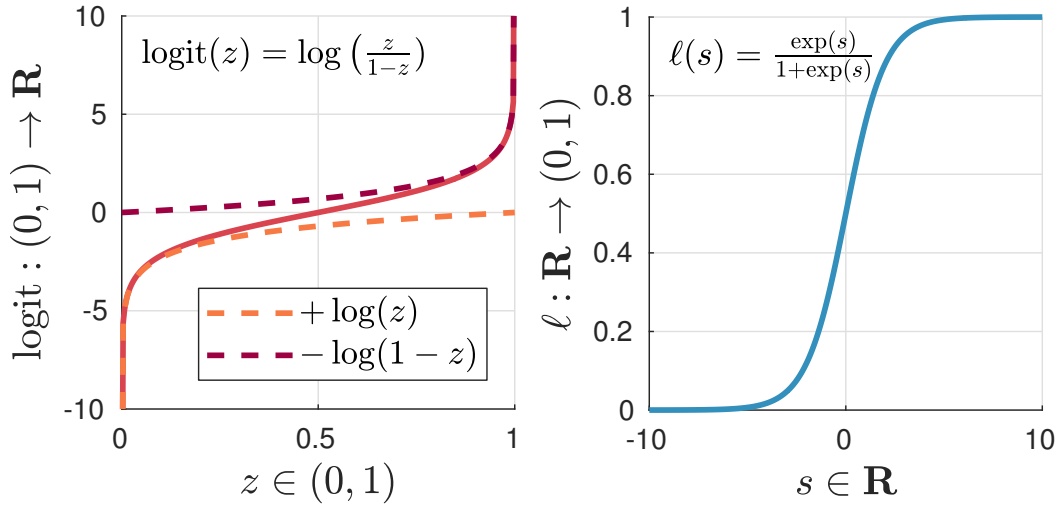
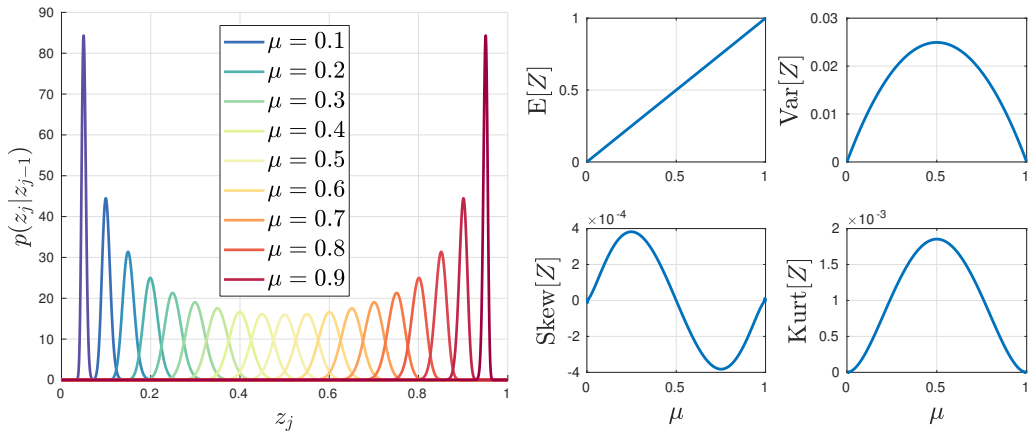


Figure 4.4: Graph of the logit function (left panel) and its inverse, the sigmoid logistic function (right panel)



(a) Probability density function of a logit-normal distribution for different values of μ . (b) Expected value, variance, skewness, and kurtosis of a logit-normal r.v. as a function of μ .

Figure 4.5: Graphical representation of the logit-normally distributed random variable $Z_j \mid (Z_{j-1} = z_{j-1}) \sim \text{logit-}\mathcal{N}(\text{logit}(\mu), \sigma^2)$ for $\sigma^2 = 0.01$, where $\mu = g_{j-1}(z_{j-1})$ denotes the (untransformed) location parameter.

that the distribution shrinks significantly when μ approaches the boundaries of $(0, 1)$.

In order to quantify such shrinkage and explore the qualitative dependence of the other important moments of the distribution on the location parameter, we also computed numerically the mean, the variance, the skewness and

the kurtosis as a function of μ . First, we notice that the expected value appears to be exactly the untransformed location parameter (top left panel in Figure 4.5b, the deviations from the identity transformation are of the order of the machine accuracy). This suggests that, unlike the log-normal case, the logit transformation generates unbiased random variations of the value predicted by the deterministic signal model (at least for small σ). In addition, the variance approaches zero at the boundaries of $(0, 1)$ and it is maximal at the interval midpoint. As for higher order moments, the fact that the skewness is positive in the left half interval and negative in the right one means that the distribution is skewed “to the boundaries”. In fact, the mode appears to be slightly closer to the boundaries with respect to the mean (not shown here). Finally, the kurtosis is maximal at the midpoint and approaches zero at the boundaries, denoting heavier tails at the center of the interval than at the boundaries (but always thinner than a normal r.v.). Note however, the small magnitude of both skewness and kurtosis indicate that these effects are very subtle, at least for the value of σ we considered.

4.4 Evaluating data assimilation methods

In this current Part I, we introduced a number of DA methods and, in this last chapter, we presented a series of technical issues which arise when applying some of those methods to real-life problems. However, we still did not say much about how one can evaluate whether a DA estimation is successful or not in practice. In the current section we address precisely such matter.

First, let us establish the notation for the current section. We here assume to be dealing with the neuron model introduced in Example 4.3. In particular, suppose we are given a set of observations $y_{0:J}$ satisfying the signal model (4.9), whose underlying true trajectory is denoted $x_{0:J}^\dagger$. Such true trajectory is assumed to be generated from the ODE model (4.6) with model parameters $\theta = \theta^\dagger$ (the true model parameters) and external input current $I_{\text{ext}}(t) = I_{\text{ext}}^\dagger(t)$. Note, however, that the following discussion is not restricted to such example of single-neuron model, but it applies to any time-dependent ODE-based state-space model (see Section 4.2).

Then, suppose we applied a given DA algorithm which provided an approximation $\widehat{p}(z|y)$ of the true posterior distribution $p(z|y)$. Note that in case of smoothing problem the posterior distribution is intended to be the smoothing distribution $p(z_{0:J}|y_{0:J})$, whereas in case of filtering problem “posterior” stands for the filtering distribution $(p(z_j|y_{0:j}))_{j=0}^J$. However, for the sake of clarity, in what follows we assume to be dealing with a smoothing problem, but the case of filtering problem is completely analogous. Recall that in case of augmented

state space, the augmented variable is $z = (x, \theta)^T$ (see Section 4.1.1). Finally, suppose that along with the approximated smoothing distribution $\widehat{p}(z_{0:J}|y_{0:J})$, the DA algorithm also provides a point estimator⁶ of the underlying dynamical signal ($\widehat{x}_{0:J}$) and a static estimator of the model parameter ($\widehat{\theta}$).

We can now detail how one can evaluate the performances of DA algorithms. Following [134, Section 2.7], we can broadly distinguish between two approaches to perform quality evaluation of a DA method. In case of **Bayesian quality assessment**, the focus is on probability distributions. In particular, to perform a Bayesian quality assessment of a DA algorithm, a comparison between the approximated posterior distribution $\widehat{p}(z_{0:J}|y_{0:J})$ and the true posterior distribution $p(z_{0:J}|y_{0:J})$ is considered. To do so, a notion of metrics on the space of probability measures should be introduced. Alternatively, at least some relevant moments of the two distribution (e.g., the mean, the covariance, etc.) need to be compared. Although rather recent and not very frequently adopted in applied DA papers, Bayesian quality assessment is adopted in evaluating 4DVAR, ExKF, EnKF and other filters in [135]. On the other hand, in the more commonly adopted case of **signal estimation quality assessment** we completely disregard the overall posterior distribution and solely compare the point estimator of the dynamical variable $\widehat{x}_{0:J}$ to the true trajectory underlying the data $x_{0:J}^\dagger$. In practice, we ask the question: how close is $\widehat{x}_{0:J}$ to $x_{0:J}^\dagger$?

Note that, although Bayesian quality assessment can be performed even when the true underlying trajectory is not available, $x_{0:J}^\dagger$ must be available in order to carry out signal estimation quality assessment. However, normally only the observations $y_{0:J}$ are available in real-life applications, whereas the underlying trajectory is typically available only in twin experiments (i.e., data assimilation experiments in which the data are artificially generated by the same model used to build the prior model).

Both the Bayesian and the signal-estimation approaches are typically applied in the data assimilation time window $\{0, \dots, J\}$ by comparing either the approximated posterior or the approximated estimator to their true counterparts. This allows one to quantify how well the unobserved trajectory can be tracked down in the data assimilation time window by the DA algorithm. However, another relevant test which is often performed in papers focused on applications is to check whether the estimated model provides reliable predictions of the “future” behaviour of the system. This is what is called checking

⁶A typical example of point estimator of a dynamical state variable x results from taking the mean of the approximated smoothing distribution as a point estimator of the hidden signal, namely $\widehat{x}_{0:J} = \int_{(\mathcal{X} \times \Theta)^{J+1}} x_{0:J} \cdot \widehat{p}(x_{0:J}, \theta_{0:J}|y_{0:J}) dx_{0:J} d\theta_{0:J}$; a notable alternative is the distribution mode $\widehat{x}_{0:J} = \operatorname{argmax}_{x_{0:J} \in \mathcal{X}^{J+1}} \int_{\Theta^{J+1}} \widehat{p}(x_{0:J}, \theta_{0:J}|y_{0:J}) d\theta_{0:J}$.

the **forecast skill** of a DA algorithm. Forecast skill quality assessment can be performed in both a Bayesian way and in a signal-estimation framework. However, we only illustrate the case of signal-estimation forecast skill because it is the only one we actually employ in our applications (see Part III).

In practice, using the endpoint of the estimated trajectory \hat{x}_J as initial condition, we can continue $\hat{x}_{0:J}$ by running the model “in the future”, using the estimated parameter $\hat{\theta}$. However, what continuation we obtain, in fact, depends on the stimulus which is presented to the system. In particular, in case of model (4.6) the stimulus is the externally applied current $I_{\text{ext}}(t)$. This means that, besides from setting the model parameter to its estimated value $\hat{\theta}$ and imposing \hat{x}_J as initial condition, the continuation is obtained by running the original deterministic model with some external current $I_{\text{ext}}^{(\text{window})}(t)$ as time-dependent input.

Here, we introduce three different continuations: **window=in**, **out1**, **out2**. First we consider the continuation in the **in-sample time window** $\hat{x}_{J:2J}^{(\text{in})}$, which corresponds to the case of injecting a stimulus identical to the one presented during the data assimilation time window, i.e. $I_{\text{ext}}^{(\text{in})}(t) \equiv I_{\text{ext}}^\dagger(t)$. Then, two different out-of sample continuations are introduced: $\hat{x}_{J:2J}^{(\text{out1})}$ corresponds to a new stimulus $I_{\text{ext}}^{(\text{out1})}(t)$ (**first out-of-sample time window**), and $\hat{x}_{J:2J}^{(\text{out2})}$ to a further different stimulus $I_{\text{ext}}^{(\text{out2})}(t)$ (**second out-of-sample time window**).

Such “predictions” are tested against the continuation of the true solution $x_{J:2J}^{\dagger(\text{window})}$ which corresponds to the suitable stimulus. To be exact, $x_{J:2J}^{\dagger(\text{in})}$, $x_{J:2J}^{\dagger(\text{out1})}$, and $x_{J:2J}^{\dagger(\text{out2})}$ are solutions of (4.6) with θ^\dagger as model parameter, x_J^\dagger as initial condition, and input current $I_{\text{ext}}^{(\text{window})} = I_{\text{ext}}^{(\text{in})}, I_{\text{ext}}^{(\text{out1})}, I_{\text{ext}}^{(\text{out2})}$, respectively. Alternatively, when the underlying true trajectory is not available, it is possible to compare the estimated continuations $\hat{x}_{J:2J}^{(\text{in})}, \hat{x}_{J:2J}^{(\text{out1})}, \hat{x}_{J:2J}^{(\text{out2})}$ to the “observations continuation” $y_{J:2J}^{(\text{window})}$ (which equals $x_{J:2J}^{\dagger(\text{window})} + \varepsilon_{J:2J}$, if $x_{J:2J}^{\dagger(\text{window})}$ is known) thanks to the measurement absolute error introduced in following section.

Note that, whereas the Bayesian quality assessment is independent of the quality of the data, the signal-estimation approach entangles a measurement of the issue of data quality and of the actual algorithm performance. Indeed, if a DA method does not give good performances in a signal-estimation framework, it might still be that the data do not contain enough information to fully characterise the unobserved signal, whereas the DA algorithm itself would have worked if the data were more informative.

In the remaining part of this chapter we illustrate a definition of performance score for DA algorithms which assesses the assimilation quality based on signal estimation. Such performance score is applied in one of the twin experiments we present in Part III, both in the data assimilation window and in a

forecast-skill evaluation setting. In addition, note that we adopt a forecast-skill approach (within the signal estimation framework) even in those applications where the performance score is not employed (see for instance Section 9.2 and Chapter 10 in Part III).

4.4.1 Performance score for algorithm evaluation

For all time windows $\text{window} = \text{in}, \text{out1}, \text{out2}$, we consider the **signal absolute error** given by the $\ell^p(\mathbb{R}^{(J+1)d_x})$ -distance between the estimated trajectory and the true one

$$d_p\left(\widehat{x}_{J:2J}^{(\text{window})}, x_{J:2J}^{\dagger(\text{window})}\right) = \left(\sum_{j=J}^{2J} \left|\widehat{x}_j^{(\text{window})} - x_j^{\dagger(\text{window})}\right|_p^p\right)^{1/p}. \quad (4.11)$$

Note that the norm $|\cdot|_p$ is the standard ℓ^p -norm on the signal space \mathbb{R}^{d_x} , so that for $p = 2$ we obtain the square root of the usual sum of squared errors (SSE).

Let us once again stress that such formula can only be applied in case of twin experiments. Indeed, a twin experiment is the only case where the true trajectory $x_{J:2J}^{\dagger(\text{window})}$ is actually known. In the general case, only the dataset $y_{J:2J}^{(\text{window})}$ is available, and the only computable quantity is the analogous $\ell^p(\mathbb{R}^{(J+1)d_y})$ -distance in the measurement space, which we name the **measurement absolute error**

$$d_p\left(H\left(\widehat{x}_{J:2J}^{(\text{window})}\right), y_{J:2J}^{(\text{window})}\right) = \left(\sum_{j=J}^{2J} \left|H\left(\widehat{x}_j^{(\text{window})}\right) - y_j^{(\text{window})}\right|_p^p\right)^{1/p}.$$

Note that the norm $|\cdot|_p$ appearing in the r.h.s. is still the usual ℓ^p -norm, but this time over the observation space \mathbb{R}^{d_y} .

However, both ℓ^p -distances we just mentioned are $[0, \infty)$ -valued, so that no upper bound quantifying the worst possible performance is available. This is not desirable because no upper bound means that we cannot assess the performance of a single run without a comparison to other results and/or a graphical inspection of the estimation. In addition, the error size is highly dependent on the size of the time window J and the model characteristics (such as the typical range of the x -components).

Nonetheless, in case of twin experiments it is still possible to normalise the distance d_p in order to obtain a sort of relative error. Indeed, given the data $y_{J:2J}^{(\text{window})}$ and the true trajectory underlying the data $x_{J:2J}^{\dagger(\text{window})}$, we define the

relative measurement error of estimate $\widehat{x}_{J:2J}^{(\text{window})}$ as the ratio

$$\begin{aligned} d_p^{(\text{rel})}(\widehat{x}_{J:2J}^{(\text{window})}, x_{J:2J}^{\dagger(\text{window})}) &:= \\ &= \frac{d_p\left(H(\widehat{x}_{J:2J}^{(\text{window})}), H(x_{J:2J}^{\dagger(\text{window})})\right)}{d_p\left(H(\widehat{x}_{J:2J}^{(\text{window})}), H(x_{J:2J}^{\dagger(\text{window})})\right) + d_p\left(H(x_{J:2J}^{\dagger(\text{window})}), y_{J:2J}^{(\text{window})}\right)}. \end{aligned} \quad (4.12)$$

Remark that in the l.h.s. we omitted the dependence on both the dataset $y_{J:2J}^{(\text{window})}$ and the observation operator H for the sake of notation compactness. In case of possible ambiguity, the complete notation should be adopted.

Remark 4.4. *The relative-error distance (4.12) essentially compares the absolute error defined in (4.11) to $d_p(H(x_{J:2J}^{\dagger(\text{window})}), y_{J:2J}^{(\text{window})})$, the mismatch between the true signal $x_{J:2J}^{\dagger(\text{window})}$ and the resulting observations $y_{J:2J}^{(\text{window})}$. If $d_p^{(\text{rel})} \approx 1$, the estimation error is much larger than the inherent mismatch between the observations $y_{J:2J}^{(\text{window})}$ and the very same trajectory $x_{J:2J}^{\dagger(\text{window})}$ that produced those data. If, on the contrary, $d_p^{(\text{rel})} \approx 0.5$ those two quantities are approximately of the same order of magnitude. Of course, $d_p^{(\text{rel})} \approx 0$ would mean that the estimation error is much smaller than the signal-observation mismatch, but this seldom occurs in our experience whereas the $d_p^{(\text{rel})} \approx 0.5$ is an attainable and favourable case. ♠*

In conclusion, the performance score we employ for signal estimation in some of the twin experiments we describe in Part III is defined as the average ($p = 1$)-relative measurement error among the three predicted trajectories considered, i.e.

$$s = \frac{1}{3} \sum_{\substack{\text{window=} \\ \text{in, out1, out2}}} d_1^{(\text{rel})}(\widehat{x}_{J:2J}^{(\text{window})}, x_{J:2J}^{\dagger(\text{window})}), \quad (4.13)$$

where $d_1^{(\text{rel})}$ is defined in (4.12). The closer this quantity is to $s = 1$, the worse the quality of the signal estimation is. On the other hand, if the performance score approaches $s = 0.5$, then the estimation quality gets better and better for all time windows. Note that we select $p = 1$ because the canonical choice $p = 2$ results in a distance d_p that penalises medium-sized deviations too much (the same for $p > 2$). In fact, for all simulation that we ran for the twin experiment illustrated in Chapter 8, the ($p = 2$)-relative errors resulted to be constantly $d_2^{(\text{rel})} \approx 1$, which means that the corresponding performance score does not recognise modest performance improvements.

As for parameters estimation, the performance evaluation is straightforward. It suffice to define the absolute and relative error in the canonical way,

namely $d^{(\text{abs})}(\widehat{\theta}, \theta^\dagger) = |\widehat{\theta} - \theta^\dagger|$ and $d^{(\text{rel})}(\widehat{\theta}, \theta^\dagger) = \frac{|\widehat{\theta} - \theta^\dagger|}{|\theta^\dagger|}$ (where the quotient is assumed to be computed component-wise in case of multidimensional parameter θ).

Part II

Neurobiology modelling

Introduction to Part II

The second part of this thesis contains the detailed description of all models of neurobiological activity which have been considered in this work. In particular, in Chapter 5 we start off by illustrating the physiology of neurons, which are the fundamental units of the nervous system characterizing essentially all animals. However, although understanding neurobiological mechanisms is essential to deal with neuron models in an meaningful application-oriented manner, driven by E. Izhikevich's work [99] we are only interested in how one can model neuronal activity with mathematical dynamical systems. For such reason, in the first Section 5.1 we solely mention the fundamental biological notions behind neuron models, so that we manage to identify the essential elements (i.e. state variables and modelling parameters) of those neural models which belong to the renowned family of Hodgkin-Huxley-type models [89]. Then, in the two remaining sections of the chapter, we describe in detail two increasingly realistic single-neuron models. Specifically, the first one is a single-compartment model with a sodium, a potassium, and a leakage ionic current (Section 5.2), whereas the one described in Section 5.3 has a plethora of ionic currents (several calcium, potassium, and sodium currents) and a complex multi-compartmental morphology [173]. For these and all models in this thesis, we first detail the mathematical model, and then give a synthetic description of its dynamics by presenting a sample trajectory for some notable values of the modelling parameters. In the last Section ?? of this chapter, we review the mathematical properties of single-neuron models, addressing in particular the well-posedness of autonomous conductance-based ODE neuron models, the techniques that allow their qualitative analysis, mention of the numerical issues involved the simulation of neural models, mention of results proving the existence and uniqueness of the solution of conductance-based neuron models given in PDE form, as well as references to the existence and analysis of travelling wave solutions.

However, the essential functional task of the central nervous system (memory, knowledge, consciousness, to mention some well-known ones) are not carried out by single neurons, but rather by groups of interconnected neurons. Indeed, without going too much into the details of these concepts, neuronal

ensembles are deemed to be the functional building blocks of the central nervous system. Therefore, in Chapter 6 we broaden our point of view in order to focus the attention on the bigger picture: i.e. on biological neural networks. We specify that the networks under consideration are biological, to stress the difference with the artificial neural networks which are nowadays heavily used in the machine learning community. First, we briefly review the digital reconstruction of the rat's neocortex portion proposed by the Blue Brain Project (Section 6.2), from which we took the above-mentioned multi-compartment neuron model. Then, in Section 6.2 we introduce a mathematical model of collective neuronal activity at a mesoscopic scale, namely a model of neural network composed of simple leaky-integrate-and-fire neurons whose dynamics is coupled via both excitatory and inhibitory synapses.

Finally, in Chapter 7 we describe different notions of distance which can be defined on the space of action potentials trains. Such distances are specifically tailored to be applied in neurobiology, and we take advantage of them in the last two numerical experiments illustrated in Part III. Such metrics are used in practice to detect spike synchrony in both couples of spike trains and larger neural populations [124].

Chapter 5

Modelling of single neurons

In this chapter we describe the main features of single-neuron modelling.

Specifically, Section 5.1 focuses on describing the building blocks of all neuron models belonging to the conductance-based family, namely the modelling of ionic currents. In this section, after a general introduction on Hodgkin–Huxley-type single-neuron models, we exemplify the notion of ionic currents by listing those ones included by the **Blue Brain Project** (BBP) in order to digitally reconstruct the second and third layer of the juvenile Wistar rats neocortex as described in [149].

Then, the following sections provide two examples of single-neuron models: Section 5.2 presents a two-dimensional neuron point model which is formally similar to the Morris–Lecar model [159], and Section 5.3 the multi-compartment single neuron model L23_PC_cADpyr229.1 which is the most frequent in the second and third layer of the Blue Brain Project microcircuit. Note that a very brief overview of the overall microcircuit composition is given in Section 6.1. In both cases, the models state variables as well as the modelling parameters are clearly listed in order to facilitate the application of the data assimilation methods described in Part I, to the neurobiological models described here.

Finally, the concluding Section 5.4 gives some reference for those theorems one invoke to establish the existence and uniqueness of the solution of single-neuron ODE models which include terms discontinuous in time.

5.1 Generalities on single neuron models

According to the seminal work of Alan L. Hodgkin and Andrew F. Huxley [89], the basic equation governing the time evolution of the membrane potential V

of a given neuron is

$$C_m \dot{V} = I_{\text{ext}} - \sum_{\text{ion}} I_{\text{ion}}.$$

In such equation, C_m is the capacitance of the neuron membrane, whereas the external current I_{ext} is typically a time-dependent function representing an input current which is experimentally injected in the neuron through a micro-electrode. In addition, the quantities $\{I_{\text{ion}}\}$ represent the set of ionic currents which pass through the neural membrane according to the opening and closing of specific ionic channels. In fact, the different electrophysiological properties observable in neural populations is due to the fact that this set of ionic species varies from neuron to neuron. Note that, by convention, the membrane potential V is assumed to be given by the electrical potential inside the cell V_{in} minus the electrical potential outside the cell V_{out} , namely $V = V_{\text{in}} - V_{\text{out}}$.

A comprehensive presentation of neural modelling is available in the textbooks [66, 113], but in what follows we only focus on the Hodgkin-Huxley-type neuron models. As we show in Section 5.1.1, in Hodgkin-Huxley's formalism ionic currents are modelled as quantities proportional to the membrane potential minus the equilibrium potential $I_{\text{ion}} = g_{\text{ion}}(V - E)$, where the conductance g_{ion} is a function of some opening and closing probabilities. This motivates why Hodgkin-Huxley-type models are also called conductance-based. However, some simpler models exist, such as the leaky integrate-and-fire (LIF) neuron model, or even coarser phase-oscillator models of the form $\dot{\varphi} = 1$ [100]. As for LIF models, we refer to the concise description given in Section 6.2. On the other hand, although the conductance-based form is now commonly accepted to account for neurons electrophysiology, more complex neuron models exist. In particular, including a description of their morphology is deemed to fundamental to describe some important effects such as action potentials back-propagation.

In fact, even though the above equation represents a single-neuron model which can accurately reproduce electrical properties of real neural cells, it certainly lacks any morphological information concerning the neuron. Nevertheless, it is rather simple to extend the model to include the spatial structure of a neuron. Indeed, a morphological conductance-based neuron model is given by the multi-compartmental ODEs system

$$C_m \dot{V}_i = I_{\text{ext}} - \sum_{\text{ion}} I_{\text{ion}} + I_{\text{cable}}, \quad \forall i = 1, \dots, N_{\text{comp}};$$

where V_i denotes the membrane potential of the i -th compartment and N_{comp} is the total number of compartments. Here, the term in I_{cable} is the one

coupling different compartments. Indeed, the transport current that results from discretizing the cable equation $\partial_t V + I(V, t) = \partial_{xx} V$ is given by

$$I_{\text{cable}} = \frac{d_i}{4R_i} \sum_k \frac{V_k - V_i}{L_{ik}^2}, \quad (5.1)$$

where the sum is over all compartments k that are adjacent to the i -th one. The modelling parameters involved in such diffusive current are: the diameter of the i -th compartment d_i , its axial resistivity R_i , and the distance between the i -th and the k -th compartment L_{ik} (i.e. the length of the compartment). Note that parameters d_i and L_{ik} are all set once the morphology of the cell is fixed.

Finally, it is possible to include synaptic interactions by simply adding a synaptic current I_{syn} which subsumes all synaptic currents coming from neurons with a post-synaptic button on the modelled neuron.

Summing all up, the most general equation that describes the time course of the membrane potential in the i -th compartment of a conductance-based model is

$$C_m \dot{V}_i = I_{\text{ext}} - \sum_{\text{ion}} I_{\text{ion}} + I_{\text{cable}} - I_{\text{syn}}, \quad \forall i = 1, \dots, N_{\text{comp}}.$$

Note that a powerful tool to numerically reproduce the activity of multi-compartmental neuron models is the Neuron simulation environment [37] available at the web page [5]. As far as we are concerned, we took advantage of such software to simulate the model described in Section 5.3 using, in particular, release 7.4 of the Neuron+Python interface (`neuron` Python module).

In remaining part of the current section, we delve into some details of the prototypical model we just introduced. Then, in the following sections two complete models of single neurons are presented: a relatively simple point model of single neuron in Section 5.2, and a detailed morphological model in Section 5.3. In conclusion, in Section 5.4 we give some pointers to the mathematical literature suitable to investigate the well-posedness of such models.

5.1.1 Ionic currents and state variables

As we said, what makes any neuron model specific is the set of ionic currents that one endow it of. Here, we describe the precise modelling of several different ionic currents, with a particular focus on those used in the BBP neuron models in the 2015 paper ‘‘Reconstruction and simulation of cortical microcircuitry’’ [149]. However, many other currents have been proposed over the years to reproduce in extreme detail neurons from different cerebral regions. Examples

include hippocampal pyramidal cells [148, 167] and cerebellar granule cells [48, 57]

In the BBP paper, a number of ionic currents was taken into account in order to model the different neurons electrophysiology, and we now list all those models of ionic currents I_{ion} that appear in the second and third layer of the BBP microcircuit. Next, we focus on what state variables are necessary to simulate each specific ionic current and how their dynamics is specified. All remarks are valid for the modelling of any ionic current in any Hodgkin–Huxley-type neuron model. After the complete list of currents and state variables, we also point out in detail what parameters are involved in the modelling of each ionic current considered in the BBP paper.

● **Sodium currents:**

- ▶ Transient sodium current [44]

$$I_{\text{Nat}}(V, m_{\text{Nat}}, h_{\text{Nat}}) = \bar{g}_{\text{Nat}} \cdot m_{\text{Nat}}^3 \cdot h_{\text{Nat}} \cdot (V - E_{\text{Na}})$$

$$I_{\text{Nat2}}(V, m_{\text{Nat2}}, h_{\text{Nat2}}) = \bar{g}_{\text{Nat2}} \cdot m_{\text{Nat2}}^3 \cdot h_{\text{Nat2}} \cdot (V - E_{\text{Na}})$$

- ▶ Persistent sodium current [146]

$$I_{\text{Nap}}(V, m_{\text{Nap}}, h_{\text{Nap}}) = \bar{g}_{\text{Nap}} \cdot m_{\text{Nap}}^3 \cdot h_{\text{Nap}} \cdot (V - E_{\text{Na}})$$

- The modelling parameters for such sodium currents include: the equilibrium potential E_{Na} and the maximal conductances \bar{g}_{Nat} , \bar{g}_{Nat2} , \bar{g}_{Nap} .

● **Potassium currents:**

- ▶ Transient potassium current [121]

$$I_{\text{Kt}}(V, m_{\text{Kt}}, h_{\text{Kt}}) = \bar{g}_{\text{Kt}} \cdot m_{\text{Kt}}^4 \cdot h_{\text{Kt}} \cdot (V - E_{\text{K}})$$

- ▶ Persistent potassium current [121]

$$I_{\text{Kp}}(V, m_{\text{Kp}}, h_{\text{Kp}}) = \bar{g}_{\text{Kp}} \cdot m_{\text{Kp}} \cdot h_{\text{Kp}} \cdot (V - E_{\text{K}})$$

- ▶ m-type potassium current [14]

$$I_{\text{M}}(V, m_{\text{M}}) = \bar{g}_{\text{M}} \cdot m_{\text{M}} \cdot (V - E_{\text{K}})$$

- ▶ Shaw-related channel-Kv3.1 potassium current [177]

$$I_{\text{Kv3.1}}(V, m_{\text{Kv3.1}}) = \bar{g}_{\text{Kv3.1}} \cdot m_{\text{Kv3.1}} \cdot (V - E_{\text{K}})$$

- ▶ d-type potassium current [189]

$$I_{Kd}(V, m_{Kd}, h_{Kd}) = \bar{g}_{Kd} \cdot m_{Kd} \cdot h_{Kd} \cdot (V - E_K)$$

- ▶ Stochastic potassium current [53]

$$I_{Kstoch}(V, N_1) = 10^{-4} \cdot [N_1]^+ \cdot \frac{\gamma_{stoch}}{\text{Area}} \cdot q_{10}^{(T-23^\circ\text{C})/10^\circ\text{K}} \cdot (V - E_K)$$

- ▶ Small conductance calcium-activated potassium current [118]

$$I_{SK}(V, z_{SK}) = \bar{g}_{SK} \cdot z_{SK} \cdot (V - E_K)$$

- The modelling parameters for the potassium currents include: the reversal potential E_K and the maximal conductances \bar{g}_{Kt} , \bar{g}_{Kp} , \bar{g}_M , $\bar{g}_{Kv3.1}$, \bar{g}_{Kd} , \bar{g}_{Kstoch} , \bar{g}_{SK} , but also the parameters of the stochastic current γ_{stoch} , Area, q_{10} , and the temperature T (expressed in degree Celsius).

● Calcium currents:

- ▶ High voltage-activated calcium current [178]

$$I_{CaHVA}(V, m_{CaHVA}, h_{CaHVA}) = \bar{g}_{CaHVA} \cdot m_{CaHVA}^2 \cdot h_{CaHVA} \cdot (V - E_{Ca})$$

- ▶ Low voltage-activated calcium current [19]

$$I_{CaLVA}(V, m_{CaLVA}, h_{CaLVA}) = \bar{g}_{CaLVA} \cdot m_{CaLVA}^2 \cdot h_{CaLVA} \cdot (V - E_{Ca})$$

- The modelling parameters for the calcium currents include: the maximal conductances \bar{g}_{CaHVA} , \bar{g}_{CaLVA} . As described below, the equilibrium potential E_{Ca} is computed as a function of the internal calcium concentration through the Nernst equation (5.7). The modelling parameter entering this expression are the external calcium concentration $[Ca^{2+}]_{out}$ and the temperature T .

● Non-specific ionic currents:

- ▶ Hyperpolarization-activated current [120]

$$I_H(V, m_H) = \bar{g}_H \cdot m_H \cdot (V - E_H)$$

- ▶ Leakage current $I_L(V)$

$$I_L(V) = \bar{g}_L \cdot (V - E_L)$$

- The modelling parameters for such non-specific currents include: the equilibrium potentials E_L and E_H and the maximal conductances g_L and g_H .

From the above list, we can notice that in the conductance-based models, all ionic currents (except the non-specific leakage current I_L) require the introduction of an activation variable m_{ion} . Also, for ion $\in \{\text{Na,t}; \text{Na,t2}; \text{Na,p}; \text{K,t}; \text{K,p}; \text{Kd}; \text{Ca,HVA}; \text{Ca,LVA}\}$ an inactivation variable h_{ion} is present too. The dynamics for such activation and inactivation variables depends on the respective membrane potential V and is given by

$$\begin{aligned}\dot{m}_{\text{ion}} &= \frac{(m_{\text{ion},\infty}(V) - m)}{\tau_{m_{\text{ion}}}(V)}, \\ \dot{h}_{\text{ion}} &= \frac{(h_{\text{ion},\infty}(V) - h)}{\tau_{h_{\text{ion}}}(V)};\end{aligned}\tag{5.2}$$

where the asymptotic value functions $m_{\text{ion},\infty}(V)$ is typically sigmoid-shaped (see the dark blue and cyan trace in Figure 5.1). In practice, it is usually a function of the transition rate functions

$$m_{\text{ion},\infty}(V) = \frac{\alpha_{m_{\text{ion}}}(V)}{\alpha_{m_{\text{ion}}}(V) + \beta_{m_{\text{ion}}}(V)},$$

where $\alpha_{m_{\text{ion}}}(V)$ and $\beta_{m_{\text{ion}}}(V)$ are given by one of the following three functions¹:

$$f^{(a)}(V) = k_1 \frac{k_2 - V}{\exp((k_2 - V)/k_3) - 1},\tag{5.3a}$$

$$f^{(b)}(V) = k_1 \exp((k_2 - V)/k_3),\tag{5.3b}$$

$$f^{(c)}(V) = k_1 \frac{1}{1 + \exp((k_2 - V)/k_3)}.\tag{5.3c}$$

On the other hand, the characteristic-time functions $\tau_{m_{\text{ion}}}(V)$ and $\tau_{h_{\text{ion}}}(V)$ typically have a Gaussian-like profiles (see the dark red and red trace in Figure 5.1) which can be obtained from the transition rate functions as

$$\tau_{m_{\text{ion}}}(V) = \frac{1}{\alpha_{m_{\text{ion}}}(V) + \beta_{m_{\text{ion}}}(V)}.$$

The discussion for $h_{\text{ion},\infty}(V)$ and $\tau_{h_{\text{ion}}}(V)$ is analogous.

¹Note that we assume that $f^{(a)}$ defined in (5.3a) is extended by continuity, i.e. $f^{(a)}(V = k_2) = k_1$

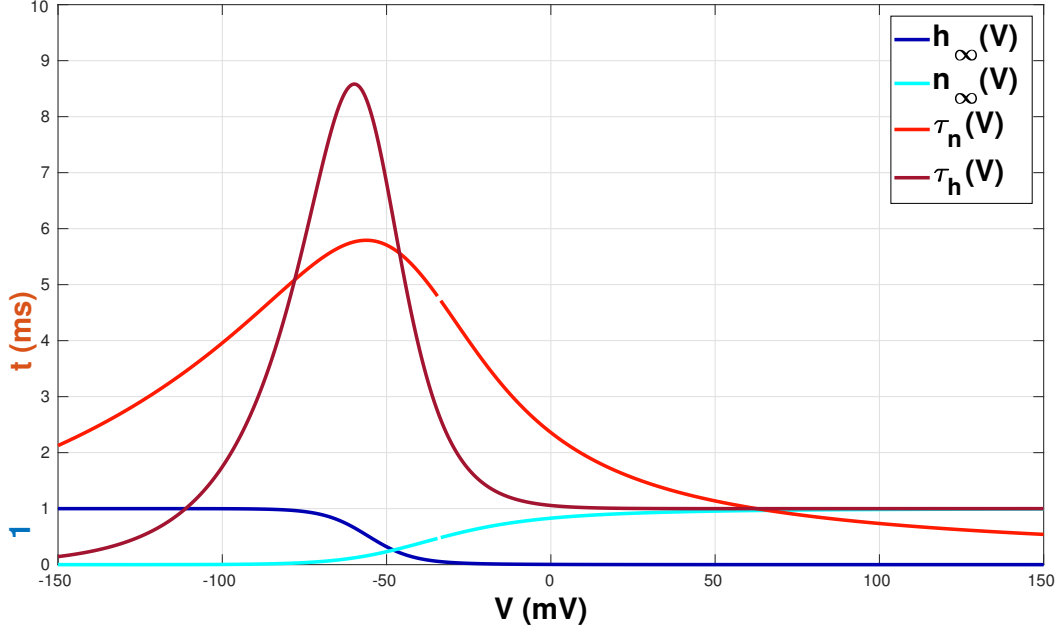


Figure 5.1: Graphs of the sigmoid-shaped asymptotic value functions for the inactivation variable of the transient sodium current $h_\infty(V)$ (dark blue line) and for the activation variable of the delayed rectifier potassium current $n_\infty(V)$ (cyan trace) appearing in the interneuron model presented in [214] and employed in a neural network model in [83]. Note that $n_\infty(V)$ is increasing because variable n is an activation variable, while $h_\infty(V)$ is decreasing since h is an inactivation variable. The red and dark red lines denote the graph of the Gaussian-shaped characteristic-time functions for the potassium current $\tau_n(V)$ and the sodium current $\tau_h(V)$, respectively. Note that their analytical expression results from the transition rate functions $\alpha_h(V) = 0.07 \exp(-\frac{V+58}{20})$, $\beta_h(V) = 1/(\exp(-0.1(V+28)) + 1)$, $\alpha_n(V) = -0.01(V+34)/(\exp(-0.1(V+34)) - 1)$, and $\beta_n(V) = 0.125 \exp(\frac{V+44}{80})$.

Besides, for some of the ionic currents² the temperature adjustment factor

$$q(T) = q_{10}^{(T-23^\circ\text{C})/10^\circ\text{K}}$$

enters the expression for the characteristic time function as a multiplicative constant. In practice, for such ionic currents the characteristic time is scaled according to the following equation

$$\tau_{m_{\text{ion}}}(V) = q(T) \frac{1}{\alpha_{m_{\text{ion}}}(V) + \beta_{m_{\text{ion}}}(V)}.$$

As a consequence, the dynamics of their activation and the inactivation variables requires the specification of the further modelling parameters q_{10} and the

²Namely, for currents I_{CaLVA} , I_{M} , I_{Kd} , I_{Kt} , I_{Kp} , I_{Kstoch} , I_{Nap} , I_{Nat} , and I_{Nat2} .

experimental temperature T in degrees Celsius.

Remark 5.1. *Note however, that the asymptotic value and the characteristic time functions are not always given via the transition rate functions α and β . In fact, there are examples where the expression for $m_{\text{ion},\infty}(V)$ and $\tau_{m_{\text{ion}}}(V)$ are given directly in one of the functional forms in (5.3). See for instance files `SKv3_1.mod` or `K_pst.mod` available at the **neocortical microcircuit collaboration** (NMC) portal [6].*

We refer to [99, Section 2.3] for a general discussion regarding the shape and the analytical form of activation and inactivation functions (on the other hand, [89] is the original reference for the original Hodgkin–Huxley neuron model). Also, consult the neuron models source codes available on the NMC portal [6] for the exact analytical form of all functions involved in the dynamics of activation and inactivation variables. ♠

Before moving to describing some actual example of single neuron models, a separate discussion should be done for the small-conductance calcium-activated potassium current I_{SK} and the stochastic potassium current I_{Kstoch} .

In fact, in the former the activation variable z_{SK} has a dynamics that depends on the internal calcium concentration $[Ca^{2+}]_{\text{in}}$. So, the presence of a I_{SK} current requires the introduction of two new state variable, whose dynamics is given by

$$\dot{z}_{\text{SK}} = \frac{z_{\infty}([Ca^{2+}]_{\text{in}}) - z_{\text{SK}}}{\tau_z}, \quad (5.4)$$

and

$$[Ca^{2+}]_{\text{in}} = -10^4 \frac{\gamma}{2F \cdot \text{depth}} I_{\text{Ca}} - \frac{[Ca^{2+}]_{\text{in}} - [Ca^{2+}]^{\text{(min)}}}{\text{decay}}. \quad (5.5)$$

The asymptotic value function for the SK current activation variable is a sigmoid-shaped Hill function [217]

$$z_{\text{SK},\infty}([Ca^{2+}]_{\text{in}}) = \frac{1}{1 + (k_1/[Ca^{2+}]_{\text{in}})^{k_2}}, \quad (5.6)$$

and the characteristic time τ_z is a constant. On the other hand, the internal calcium concentration depends on the total calcium current I_{Ca} (given by the sum of all calcium currents included in the model), the percentage of free calcium γ (not buffered, [*sic*]), the Faraday constant F , the “depth of an imaginary sub-membrane cell” [51] $\text{depth} > 0$, the minimum internal calcium concentration $[Ca^{2+}]^{\text{(min)}}$ and the reciprocal of the removal rate of calcium ions $\text{decay} > 0$.

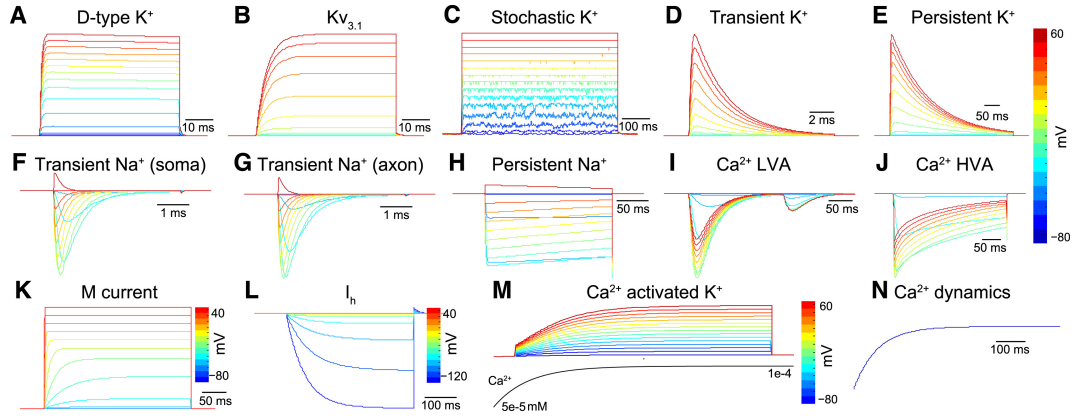


Figure 5.2: Ionic currents dynamics for different values of the membrane potential. Source: [149]

Since the internal calcium concentration is explicitly modelled, the reversal potential of calcium currents E_{Ca} is consequently adapted according to the Nernst equation

$$E_{Ca} = k' \frac{T_{kelv} \cdot R}{2F} \log \left(\frac{[Ca^{2+}]_{out}}{[Ca^{2+}]_{in}} \right), \quad (5.7)$$

where R is the universal gas constant, $T_{kelv} = T + 273.15^\circ$ is the temperature in degree Kelvin, $[Ca^{2+}]_{out}$ is the calcium concentration outside the cell, and $k' = 1000$ for E_{Ca} measured in millivolts [52].

Finally, the stochastic potassium current I_{Kstoch} depends on the random process $N_1(t)$, which represents the number of opened ionic channels and acts as a further state variable. The law governing the dynamics of the stochastic potassium state variable N_1 is given by

$$\left\{ \begin{array}{l} N_0(t + \Delta t) = [N_0(t) - n_{01}(t) + n_{10}(t)]^+ \\ N_1(t + \Delta t) = N_{tot} - N_0(t) \\ n_{01}(t) \sim \text{Binomial}(P_\alpha, N_0(t)) \\ P_\alpha = [\alpha_{Kstoch}(V) \cdot \Delta t]^+ \\ n_{10}(t) \sim \text{Binomial}(P_\beta, N_1(t)) \\ P_\beta = [\beta_{Kstoch}(V) \cdot \Delta t]^+ \\ N_{tot} = \lfloor \bar{g}_{Kstoch} \frac{\text{Area}}{\gamma_{stoch}} + 0.5 \rfloor. \end{array} \right.$$

The random variable N_0 represents the number of ionic channels that are closed, $n_{01}(t)$ and $n_{10}(t)$ are stochastic integer-valued processes representing the number of channels moving from the closed state to the opened state at time t , and vice-versa. Terms P_α and P_β represent the probability that one channel opens or closes, respectively. In fact, P_α and P_β are functions of the

transition rate functions $\alpha_{\text{Kstoch}}(V)$ and $\beta_{\text{Kstoch}}(V)$. Finally, Δt is the numerical time step and N_{tot} is the total number of ion channels, which can be computed from the maximal conductance \bar{g}_{Kstoch} and the parameters Area and γ_{stoch} .

In conclusion, we refer to Figure 5.2 for a sample time course of all ionic currents described in this section. In particular, note the difference between the profile of transient and persistent currents: persistent currents maintain an approximately constant flow of ions if the membrane potential V is kept at a constant value, whereas transient currents decay to zero after an initial spike of ion flow. Finally, Table 5.3 summarises all state variables and modelling parameters involved in the mathematical representation of each ionic current we mentioned so far.

In the remaining part of this chapter, we illustrate in detail two different models of single neuron.

5.2 Toy model for a single neuron

First, let us start off by presenting a single-compartment two-dimensional neuronal model which is somewhat similar to the Morris–Lecar model [159]. Indeed, it has the same number of ionic currents, the same number of state variables, similar dynamics, and similar bifurcation diagrams [99, Chapter 4]. The ODEs system representing such neuron is

$$\begin{cases} C\dot{V} &= -\bar{g}_{\text{K}}a(V - E_{\text{K}}) - \bar{g}_{\text{Na}}b_{\infty}(V)(V - E_{\text{Na}}) \\ &\quad -\bar{g}_{\text{L}}(V - E_{\text{L}}) + I_{\text{ext}}(t) \\ \tau_a\dot{a} &= a_{\infty}(V) - a, \end{cases}, \quad t \in [0, T_f]; \quad (5.8)$$

where V is the membrane potential and the ionic-channel activation variable a represents the opening probability of the transmembrane potassium channels. Parameters of the model include the membrane capacity C and the time scale constant τ_a , the maximal conductance of the potassium, the sodium, and the leakage ionic currents (\bar{g}_{K} , \bar{g}_{Na} and \bar{g}_{L} respectively), and the corresponding equilibrium potentials (E_{K} , E_{Na} and E_{L}). Furthermore, the asymptotic value function for such model is a particular instance of (5.3c), namely

$$a_{\infty}(V) = 1 / \left[1 + \exp \left((V_{1/2}^{(a)} - V) / K^{(a)} \right) \right], \quad (5.9)$$

where $K^{(a)}$ is a steepness parameter and $V_{1/2}^{(a)}$ is such that $a_{\infty}(V_{1/2}^{(a)}) = 0.5$. The function $b_{\infty}(V)$ satisfies a relation analogous to (5.9) with corresponding parameters $V_{1/2}^{(b)}$ and $K^{(b)}$. The input $I_{\text{ext}}(t)$ represents a preassigned time-dependent externally-applied current, which we assume to be a piecewise con-

Ion current	State variables		Parameters
Sodium currents			
I_{Nat}	m_{Nat}	h_{Nat}	$\bar{g}_{\text{Nat}}, E_{\text{Na}}, q_{10}, T$
I_{Nat2}	m_{Nat2}	h_{Nat2}	$\bar{g}_{\text{Nat2}}, E_{\text{Na}}, q_{10}, T$
I_{Nap}	m_{Nap}	h_{Nap}	$\bar{g}_{\text{Nap}}, E_{\text{Na}}, q_{10}, T$
Potassium currents			
I_{Kt}	m_{Kt}	h_{Kt}	$\bar{g}_{\text{Kt}}, E_{\text{K}}, q_{10}, T$
I_{Kp}	m_{Kp}	h_{Kp}	$\bar{g}_{\text{Kp}}, E_{\text{K}}, q_{10}, T$
I_{M}	m_{M}		$\bar{g}_{\text{M}}, E_{\text{K}}, q_{10}, T$
$I_{\text{Kv3.1}}$	$m_{\text{Kv3.1}}$		$\bar{g}_{\text{Kv3.1}}, E_{\text{K}}$
I_{Kd}	m_{Kd}	h_{Kd}	$\bar{g}_{\text{Kd}}, E_{\text{K}}, q_{10}, T$
I_{Kstoch}	N_1		$\bar{g}_{\text{Kstoch}}, E_{\text{K}}, q_{10}, T$
Calcium and calcium-activated currents			
I_{CaHVA}	m_{CaHVA}	h_{CaHVA}	$\bar{g}_{\text{CaHVA}}, [Ca^{2+}]_{\text{out}}$
I_{CaLVA}	m_{CaLVA}	h_{CaLVA}	$\bar{g}_{\text{CaLVA}}, [Ca^{2+}]_{\text{out}}, q_{10}, T$
I_{SK}	z_{SK}		$\bar{g}_{\text{SK}}, E_{\text{K}}$
	$[Ca^{2+}]_{\text{in}}$		$\gamma, \text{decay}, \text{depth}, [Ca^{2+}]^{(\text{min})}$
Aspecific currents			
I_{H}	m_{H}		$\bar{g}_{\text{H}}, E_{\text{H}}$
I_{L}			$\bar{g}_{\text{L}}, E_{\text{L}}$

Table 5.3: Table listing all state variables and parameters involved in the modelling of a each ionic current (mechanism) considered in [149]. State variable $[Ca^{2+}]_{\text{in}}$ and the relative parameters are present for all calcium-related currents. Parameters with the same name across different currents assume the same value in a given compartment.

stant function given by

$$I_{\text{ext}}(t) = \begin{cases} I_1 & t \in [0, T^{(1)}) \\ I_i & t \in [T^{(i-1)}, T^{(i)}), \quad i = 2, \dots, i_{\text{max}} \\ I_{i_{\text{max}}} & t \in [T^{(i_{\text{max}})}, T_f], \end{cases} \quad (5.10)$$

where T_f is the time horizon of model (5.8) and the set of the jump times $\{T^{(i)}\}_{i=1}^{i_{\text{max}}}$ constitutes a Poisson process of rate λ (i.e. the expected interspike interval is $\mathbb{E}[T^{(i+1)} - T^{(i)}] = 1/\lambda$ for all $i = 1, \dots, i_{\text{max}} - 1$). The corresponding step-current values $\{I_i\}_{i=1}^{i_{\text{max}}}$ are modelled as an i.i.d. random sequence with uniform distribution over the current range $[I_{\text{low}}, I_{\text{upp}}]$.

θ^\dagger	True value	Unit
\bar{g}_{Na}	20	mS cm ⁻²
E_{Na}	60	mV
\bar{g}_{K}	10	mS cm ⁻²
E_{K}	-90	mV
\bar{g}_{L}	8	mS cm ⁻²
E_{L}	-78	mV
$V_{1/2}^{(b)}$	-20	mV
$K^{(b)}$	15	mV
$V_{1/2}^{(a)}$	-45	mV
$K^{(a)}$	5	mV
τ_a	1	ms
C	1	μF cm ⁻²

Table 5.4: Model L23_PC_cADpyr229_1: true parameter values and unit measure

5.2.1 Sample toy model trajectory

In order to investigate what kind of responses such model produces, it is necessary to fix all model parameters. We firstly fix the external current $I_{\text{ext}}(t)$ by drawing a single instance of $\{T^{(i)}\}_{i=1}^{i_{\text{max}}}$ and $\{I_i\}_{i=1}^{i_{\text{max}}}$ with $\lambda = 1 \text{ ms}^{-1}$, $I_{\text{low}} = -5 \mu\text{A cm}^{-2}$, and $I_{\text{upp}} = 40 \mu\text{A cm}^{-2}$. Then, we also set the true parameter values to be the values listed in Table 5.4. Finally, we use a fourth-order Runge-Kutta method to solve the system of ordinary differential equations (5.8) over the time mesh $t_{0:J} = \{t_j\}_{j=0}^J$, where $t_j = j\Delta t$, $\Delta t = 0.01 \text{ ms}$, and $T_f = t_J = 500 \text{ ms}$. The initial condition is set to be $(V(0), a(0))^T = (V_0^\dagger, a_0^\dagger)^T$ where $V_0^\dagger = -64 \text{ mV}$, and $a_0^\dagger = a_\infty(V_0^\dagger)$. For future references, we call **true trajectory** the resulting approximate solution and write $x_{0:J}^\dagger = \{(V_j^\dagger, a_j^\dagger)^T\}_{j=0}^J$.

To produce a dataset usable for twin experiments (such as the twin experiment we describe in Chapter 8), we also need to generate some measurements associated to the true trajectory $x_{0:J}^\dagger$. In this case, we assume no measurement occurs at $j = 0$. Hence, the dataset $y_{1:J} = \{y_{j+1}\}_{j=0}^{J-1}$ is produced by drawing a single realization of the measurement noise $\varepsilon_{1:J} = \{\varepsilon_{j+1}\}_{j=0}^{J-1}$ with standard deviation $\Gamma^{1/2} = 1 \text{ mV}$, and setting

$$y_{j+1} = V_{j+1}^\dagger + \varepsilon_{j+1}, \quad j = 0, \dots, J-1.$$

The first 200 ms of the simulated dataset $y_{1:J}$ are represented in Figure 5.5, along with the underlying true trajectory $x_{0:J}^\dagger$ and the stimulus I_{ext} which

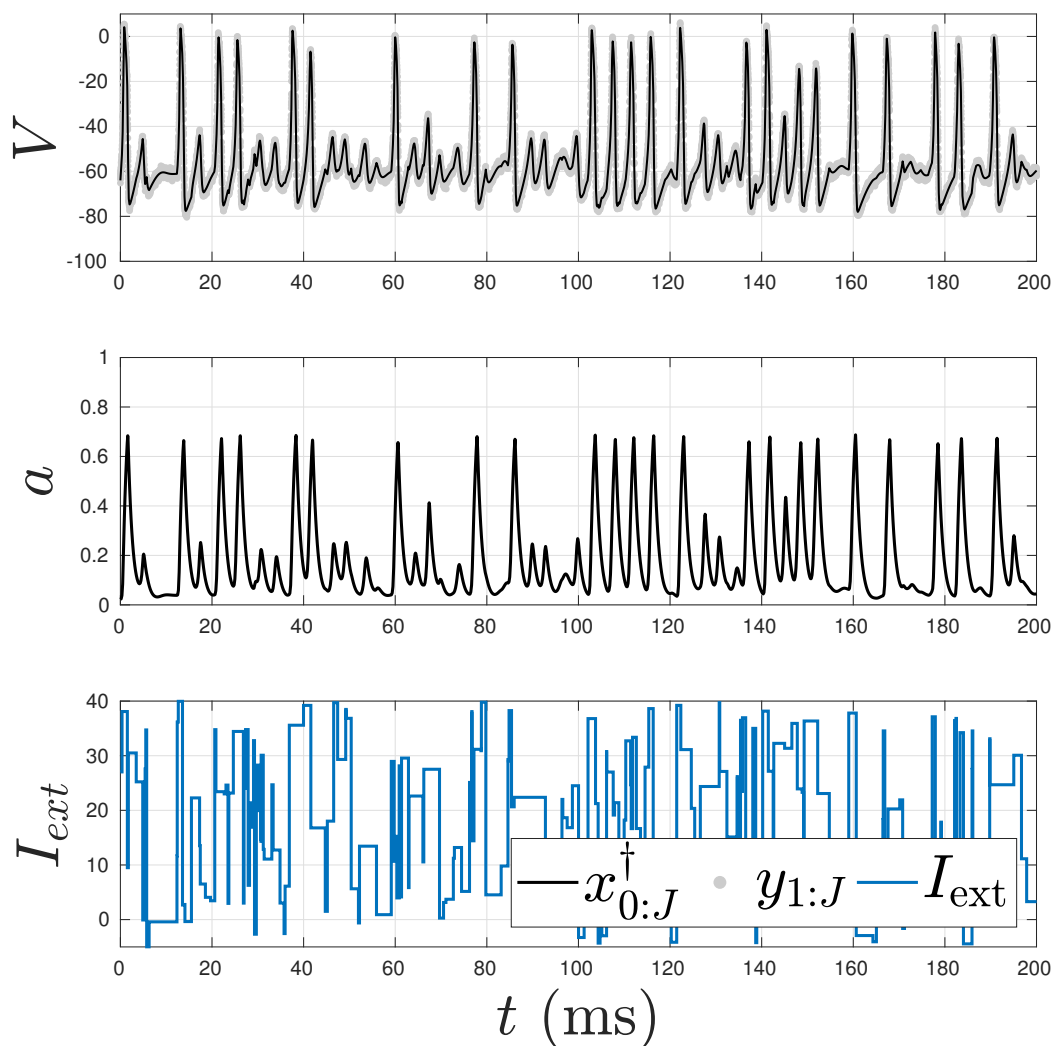


Figure 5.5: Sample response of model (5.8) with parameter values as in Table 5.4 to an input current of the form (5.10) with $\lambda = 1 \text{ ms}^{-1}$ and $[I_{\text{low}}, I_{\text{upp}}] = [-5 \mu\text{A cm}^{-2}, 40 \mu\text{A cm}^{-2}]$. The true trajectory $x_{0:J}^\dagger$ (black plain line) is obtained by solving the model with a Runge-Kutta method of the fourth order with computational step of $\Delta t = 0.01 \text{ ms}$. The grey dots in the top panel denote the artificial dataset $y_{1:J}$ obtained by adding a Gaussian noise with 1 mV-wide standard deviation to the V -component of the true solution.

produces such neural response.

Remark 5.2. Note that what we just described implies the mathematically-detailed discretization procedure introduced in Section 4.2. In particular, such procedure (which serves to define the discrete signal and data state-space mod-

els) is instantiated in Example 4.3 where it is applied to the very same model (5.8) defined in the current section. ♠

In the following section, we describe a realistic model of morphologically-accurate single neuron.

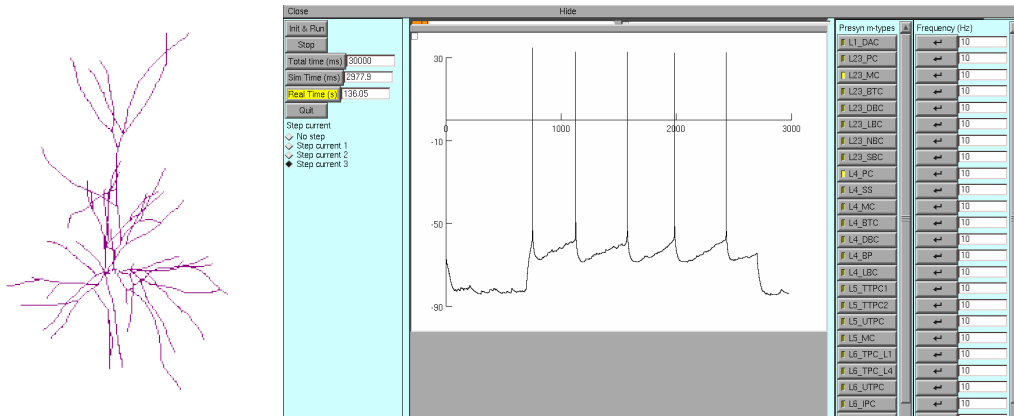
5.3 Realistic model from the Blue Brain Project

In this section, we describe in detail the model for pyramidal cells with soma lying in the second and third neocortical layer L23_PC_cADpyr229_1 developed and used in the neocortical microcircuit depicted in [149]. In particular, in Section 5.3.1 the ODEs for each compartment class is stated, in Section 5.3.2 all model parameters are listed, and in Section 5.3.3 we present a sample response of the model to a step depolarizing current.

First, let us start off by some generalities on multi-compartment models and on the Neuron simulation environment [86]. The mathematical model for any neuron model considered in the BBP neocortical microcircuit consists in a multi-compartment conductance-based model. The morphology for such models (i.e. the compartmental structure of the model) is derived from experimental data obtained by the Markram Lab, with the addition of some data-based variability. Such variability reflects the jittering branch angle and section length statistics obtained by comparing the morphology of different neurons belonging to the same neuron class. Also, all models from the Blue Brain project are implemented in the Neuron simulation environment [37, 86]. We highlight that, for the ease of use for experimental modellers, Neuron introduces a top-layer compartment class called **section**. A section is a compartment with homogeneous geometrical and electrical properties. However, it is sometimes desirable to split a given section in shorter **segments**, in order to improve computational accuracy. From a mathematical point of view, each segment is a compartment, whereas a section is nothing but a collection of segments with homogeneous morphological properties. In what follows, whenever we talk of compartments, we refer to the segments of the model.

In the twin experiment setting described in Chapter 9, we consider the model proposed in [149] for the layer 2 and layer 3 (L23) continuous-adapting (cAD) pyramidal cells (PC). This model was selected because it is the most common in the layer two and three of the microcircuit (which are regarded as one in the neural network modelling), as showed in Table 6.4. For further details on the models nomenclature and on the modelling process see Section 6.1.

In order to fix once for all the model in exam, we consider the morphology of clone L23_PC_cADpyr229_1 depicted in Figure 5.6a. Such morphology can



(a) Neuron morphology for clone L23_PC_cADpyr229_1. (b) Neuron graphical user interface (GUI) screenshot showing a sample somatic potential time course with both depolarizing step input current and some synaptic inputs.

Figure 5.6: Representation of model L23_PC_cADpyr229_1 for layer 2 and 3 pyramidal cells. Produced with Neuron using the source code for this model available at the NMC portal.

be downloaded from the neocortical microcircuit collaboration portal (NMC portal) [6]. We refer to [173] for a detailed description of the content of the portal and its use. Figure 5.6b presents a sample somatic potential time course of model L23_PC_cADpyr229_1, plotted in the Neuron graphical user interface (GUI). The input producing that trace are: a depolarizing step input current, inhibitory synaptic inputs coming from layer 2-3 Martinotti cells, and excitatory synaptic inputs originated from layer 4 pyramidal cells.

We now examine in detail the mathematical model for layer 2 and 3 pyramidal cells.

5.3.1 Mathematical model for L23_PC_cADpyr229_1

Model L23_PC_cADpyr229_1 consists in a set of ODEs for each compartment considered in the morphology. In all models considered in [149], the form of the equations depends on the type of compartment. In particular, for pyramidal cell models the authors considered a single somatic compartment, two axonal compartments, and several basal and apical dendritic compartments.

Here, we write down the differential equation for the membrane potential of each of these four classes.

- 1 somatic section, for a total of 1 segment

$$(\text{soma}) \left\{ \begin{array}{l} C_m \dot{V}^{(\text{soma})} = + I_{\text{ext}}(t) - I_{\text{syn}} + I_{\text{cable}} \\ \quad - I_{\text{Nat2}} - I_{\text{Kv3.1}} - I_{\text{SK}} \\ \quad - I_{\text{CaLVA}} - I_{\text{CaHVA}} \\ \quad - I_{\text{H}} - I_{\text{L}} \end{array} \right. ; \quad (5.11)$$

- 2 axonal sections, for a total of 2 segments

$$(\text{axonal}) \left\{ \begin{array}{l} C_m \dot{V}_i^{(\text{axon})} = - I_{\text{syn}} + I_{\text{cable}} \\ \quad - I_{\text{Nat}} - I_{\text{Nap}} \\ \quad - I_{\text{Kt}} - I_{\text{Kp}} - I_{\text{Kv3.1}} - I_{\text{SK}}, \quad \forall i = 1, 2; \\ \quad - I_{\text{CaLVA}} - I_{\text{CaHVA}} \\ \quad - I_{\text{H}} - I_{\text{L}} \end{array} \right. \quad (5.12)$$

- 23 apical sections, for a total of 109 segments

$$(\text{apical}) \left\{ \begin{array}{l} C_m \dot{V}_i^{(\text{apic})} = - I_{\text{syn}} + I_{\text{cable}} \\ \quad - I_{\text{Nat2}} - I_{\text{M}} - I_{\text{Kv3.1}}, \quad \forall i = 1, \dots, 109; \\ \quad - I_{\text{H}} - I_{\text{L}} \end{array} \right. \quad (5.13)$$

- 66 basal dendritic sections, for a total of 200 segments

$$(\text{dendritic}) \left\{ \begin{array}{l} C_m \dot{V}_i^{(\text{dend})} = - I_{\text{syn}} + I_{\text{cable}}, \quad \forall i = 1, \dots, 200. \\ \quad - I_{\text{H}} - I_{\text{L}} \end{array} \right. \quad (5.14)$$

Note that the exact form of all ionic currents is the one given in Section 5.1.1. In practice, all ionic currents in each compartment (except for I_{L}) have some further state variable whose dynamics is given in (5.2), (5.4) or (5.5). As a consequence, in the right hand side of (5.12), (5.13), and (5.14), all ionic currents depend on the state variables of the respective compartment. For instance, the complete membrane potential ODE for apical dendrites writes

$$\left\{ \begin{array}{l} C_m^{(\text{apic})} \dot{V}_i^{(\text{apic})} = - I_{\text{syn}} + I_{\text{cable}} \\ \quad - I_{\text{Nat2}}^{(\text{apic})} (V_i^{(\text{apic})}, m_{\text{Nat2},i}^{(\text{apic})}, h_{\text{Nat2},i}^{(\text{apic})}) \\ \quad - I_{\text{M}}^{(\text{apic})} (V_i^{(\text{apic})}, m_{\text{M},i}^{(\text{apic})}) - I_{\text{Kv3.1}}^{(\text{apic})} (V_i^{(\text{apic})}, m_{\text{Kv3.1},i}^{(\text{apic})}) \end{array} \right. ,$$

for $i = 1, \dots, 109$. Notice the i subscript that identifies all compartment-specific state variables appearing in the right hand side of such equation. Also,

note that $C_m^{(\text{apic})}$ is the value of the membrane capacitance for apical dendrites, and that we added the “(apic)” superscript to all ionic currents ($I_{\text{ion}}^{(\text{apic})}$) to stress the fact that modelling parameters appearing in their expression (such as maximal conductances) are section-dependent. Details regarding modelling parameters are given in the following section.

Writing down the differential equations for the activation/inactivation variables, the complete model for apical compartments turns out to be

$$\left\{ \begin{array}{l} C_m^{(\text{apic})} \dot{V}_i^{(\text{apic})} = - I_{\text{syn}} + I_{\text{cable}} \\ \quad - I_{\text{Nat2}}^{(\text{apic})} (V_i^{(\text{apic})}, m_{\text{Nat2},i}^{(\text{apic})}, h_{\text{Nat2},i}^{(\text{apic})}) \\ \quad - I_{\text{M}}^{(\text{apic})} (V_i^{(\text{apic})}, m_{\text{M},i}^{(\text{apic})}) - I_{\text{Kv3.1}}^{(\text{apic})} (V_i^{(\text{apic})}, m_{\text{Kv3.1},i}^{(\text{apic})}) \\ \quad - I_{\text{H}}^{(\text{apic})} (V_i^{(\text{apic})}, m_{\text{H},i}^{(\text{soma})}) - I_{\text{L}}^{(\text{apic})} (V_i^{(\text{apic})}) \\ \dot{m}_{\text{Nat2},i}^{(\text{apic})} = \left(m_{\text{Nat2},\infty}^{(\text{apic})} (V_i^{(\text{apic})}) - m_{\text{Nat2},i}^{(\text{apic})} \right) / \tau_{m_{\text{Nat2}}} (V_i^{(\text{apic})}) \\ \dot{h}_{\text{Nat2},i}^{(\text{apic})} = \left(h_{\text{Nat2},\infty}^{(\text{apic})} (V_i^{(\text{apic})}) - h_{\text{Nat2},i}^{(\text{apic})} \right) / \tau_{h_{\text{Nat2}}} (V_i^{(\text{apic})}) \\ \dot{m}_{\text{M},i}^{(\text{apic})} = \left(m_{\text{M},\infty}^{(\text{apic})} (V_i^{(\text{apic})}) - m_{\text{M},i}^{(\text{apic})} \right) / \tau_{m_{\text{M}}} (V_i^{(\text{apic})}) \\ \dot{m}_{\text{Kv3.1},i}^{(\text{apic})} = \left(m_{\text{Kv3.1},\infty}^{(\text{apic})} (V_i^{(\text{apic})}) - m_{\text{Kv3.1},i}^{(\text{apic})} \right) / \tau_{m_{\text{Kv3.1}}} (V_i^{(\text{apic})}) \\ \dot{m}_{\text{H},i}^{(\text{soma})} = \left(m_{\text{H},\infty}^{(\text{apic})} (V_i^{(\text{apic})}) - m_{\text{H},i}^{(\text{soma})} \right) / \tau_{m_{\text{H}}} (V_i^{(\text{apic})}) \end{array} \right. ,$$

for $i = 1, \dots, 109$. Moreover, the model for the i -th segment (i.e. compartment) is coupled to the adjacent segments of the model through the cable currents I_{cable} . In fact, the membrane potential of the segments that are connected to the i -th segment appears in the r.h.s. of (5.1). Finally, the synaptic current I_{syn} links a given segment to other cells in the microcircuit model.

In conclusion, considering all compartments the mathematical model for L23_PC_cADpyr229_1 has 92 sections (1 somatic section, 2 axonal sections, 23 apical sections, 66 dendritic sections) corresponding to 312 segments or compartments (1 somatic segment, 2 axonal segments, 109 apical segments, and 200 dendritic segments), for a total of 1097 state variables (11 state variables per somatic compartment, 16 state variables per axonal compartment, 6 state variables per apical compartment, and 2 state variables per dendritic compartment), and many model parameters.

We now discuss exactly which modelling parameters enter such model equations.

5.3.2 Model parameters

Let us investigate the details about the parameters involved in the modelling of pyramidal cells of the second and third cortical layer. Once we assign the neuron model morphology (i.e., the spatial position, dimension and orientation of all neuron sections), we fix at once all parameters d_i and L_{ik} appearing in the expression for I_{cable} (5.1) in all compartments. Besides, we assume that all the functions and parameters involved in the dynamics of the activation and inactivation variables are fixed (namely, all k_i appearing in (5.3) and (5.6), and τ_z in (5.4)).

Then, we are left with all parameters C_m and R , the reversal potentials E_{ion} , the maximal conductances of all ionic currents \bar{g}_{ion} , the temperature adjustment factor parameters q_{10} and T (the temperature in degree Celsius), along with all parameters involved in the internal calcium concentration dynamics (5.5). Note that the external calcium concentration $[Ca^{2+}]_{\text{out}}$ (which, according to (5.7), is required to compute the calcium equilibrium potential E_{Ca}) is considered constant and then it is a further modelling parameter.

All the parameters that appear in the neuron model L23_PC_cADpyr229_1 are listed in Table 5.7. The parameter values used in [149] appear in the second, third, fourth or fifth column, according to the compartment class. Note, in fact, that modelling parameters assume the same value in the segments belonging to a given class (i.e. somatic, axonal, apical, and basal segments), but may differ from one class to the other. Blank spaces indicate the absence of a parameter for a specific section class (e.g. there is no E_{Na} in the Basal column because no sodium mechanism is included in the basal dendrites model). Also, the last column indicates whether a parameter was manually fixed by the modellers (\times) or if it was set as a free parameter (\checkmark), and then computed by the multi-objective optimisation procedure proposed in [60, 61] and refined into a open-source tool for the community of neurobiology modellers in [203].

5.3.3 Sample dynamics and artificial dataset

Let us consider the model given by (5.11), (5.12), (5.13), and (5.14) with values of the modelling parameters to be those listed in Table 5.7. What is a sample trajectory of the resulting model? How can we generate an artificial dataset in order to perform twin experiments on such model?

First of all, we need to fix the input external current $I_{\text{ext}}(t)$ that appears in (5.11). Our choice of input current reflects the Neuron code for model L23_PC_cADpyr229_1 available on the NMC portal, where three input currents with different depolarizing amplitude are considered. Namely, three amplitudes `window=in,out1,out2`, which correspond to 1.5, 2, and 2.5 times the

Parameter	Somatic	Axonal	Apical	Basal	Unit	Free
Passive electrical properties						
C_m	1	1	2	2	$\mu\text{F}/\text{cm}^2$	\times
R	100	100	100	100	$\Omega \text{ cm}$	\times
Equilibrium potentials						
E_{Na}	50	50	50		mV	\times
E_{K}	-85	-85	-85		mV	\times
E_{H}	-45	-45	-45	-45	mV	\times
E_{L}	-75	-75	-75	-75	mV	\times
Maximal ionic conductances						
\bar{g}_{Nat}		3429.725			mS/cm^2	\checkmark
$\bar{g}_{\text{Nat}2}$	926.705		12.009		mS/cm^2	\checkmark
\bar{g}_{Nap}		9.803			mS/cm^2	\checkmark
\bar{g}_{Kt}		1.035			mS/cm^2	\checkmark
\bar{g}_{Kp}		959.296			mS/cm^2	\checkmark
\bar{g}_{M}			0.74		mS/cm^2	\checkmark
$\bar{g}_{\text{Kv}3.1}$	102.517	94.971	0.513		mS/cm^2	\checkmark
\bar{g}_{SK}	99.433	8.085			mS/cm^2	\checkmark
\bar{g}_{CaHVA}	0.374	0.306			mS/cm^2	\checkmark
\bar{g}_{CaLVA}	0.778	0.05			mS/cm^2	\checkmark
\bar{g}_{H}	0.08	0.08	$0.08g(y)$	0.08	mS/cm^2	\times
\bar{g}_{L}	0.03	0.03	0.03	0.03	mS/cm^2	\times
$[\text{Ca}^{2+}]_{\text{in}}$ dynamics						
γ	0.000533	0.016713			1	\checkmark
decay	342.544232	384.114655			ms	\checkmark
depth	0.1	0.1			μm	\times
$[\text{Ca}^{2+}]^{\text{(min)}}$	0.0001	0.0001			mM	\times
Parameters for E_{Ca} and temperature adjustment factor						
q_{10}	2.3	2.3	2.3		1	\times
T	34	34	34		$^{\circ}\text{C}$	\times
$[\text{Ca}^{2+}]_{\text{out}}$	2	2			mM	\times

Table 5.7: Parameters of model L23_PC_cADpyr229_1 as set in [149]. Parameter values in somatic, axonal, apical and basal compartments are listed in the 2nd, 3rd, 4th and 5th column, respectively. The value of \bar{g}_{H} is exponentially distributed in the apical sections (i.e. $g(y) = -0.8696 + 2.087 \cdot \exp(y/0.0031)$ where y is the distance of the section from the soma. The sixth column shows the unit of measure for each parameter. Finally, a \times sign in the last column means that the value was manually set by the modellers, whereas a \checkmark sign indicates that the value was numerically evaluated by the multi-objective evolutionary algorithm discussed in [60].

rheobase³, respectively. All three inputs are given by the superimposition of a hyperpolarizing step current (necessary to hold the cell around the resting potential of approximately $V = -85$ mV), and a depolarizing step current. The hyperpolarizing current lasts 3000 ms (the entire duration of the experiment), while the depolarizing current triggers after 700 ms and lasts 2000 ms. In formulas

$$I_{\text{ext}}^{(\text{window})}(t) = \begin{cases} I_{\text{hyp}} & t \in [0, 700\text{ms}) \\ I_{\text{hyp}} + I_{\text{dep}}^{(\text{window})} & t \in [700\text{ms}, 2700\text{ms}) \\ I_{\text{hyp}} & t \in [2700\text{ms}, 3000\text{ms}] \end{cases}, \quad (5.15)$$

where $I_{\text{hyp}} = -0.071777$ nA for all three step currents. The three inputs only differ for the depolarizing current amplitude: for `window=in` the depolarizing current amplitude is $I_{\text{dep}}^{(\text{in})} = 0.1626792$ nA; for `window=out1` $I_{\text{dep}}^{(\text{out1})} = 0.1762358$ nA; and for `window=out2` $I_{\text{dep}}^{(\text{out2})} = 0.1897924$ nA. At this point, the time-dependent input current is fixed.

Now, the artificial dataset is generated in the following way. First, the Neuron+Python implementation of model `L23_PC_cADpyr229_1` is numerically solved using the backward-Euler solver in a 3000 ms time window with input current $I_{\text{ext}}^{(\text{in})}$. The non-adaptive time step is $\Delta t = 0.25$ ms and the initial condition is $V(t_0 = 0 \text{ ms}) = -65$ mV, $[Ca^{2+}]_{\text{in}}(t_0) = 5e-5$ mM, whereas the activation and inactivation variables are initialised to their asymptotic value, i.e. $m_{\text{ion}}(t_0) = m_{\text{ion},\infty}(-65\text{mV})$, and analogously for $h_{\text{ion}}(t_0)$. We take the resulting trajectory⁴ $x_{0:J}^\dagger = \{x_j^\dagger\}_{j=0}^J$ as the **true discrete trajectory** underlying the observations in the **data assimilation time window** [0 ms, 3000 ms]. Note that according to such notation, x_j^\dagger denotes the value of the true solution of the neuron model at time $t_j = j\Delta t$.

Then, the observable dataset is generated by simply adding a Gaussian measurement noise with standard deviation $\Gamma^{1/2} = 1$ mV to the membrane potential. Mimicking the notation established in Section 4.2, we write that the observation operator $H : \mathcal{Z} \rightarrow \mathcal{Y}$ is the projection on the first component (the membrane potential), so that $Y_j = H(x_j^\dagger) + \varepsilon_j = V(t_j) + \varepsilon_j$, where the noise process is an i.i.d sequence with $\varepsilon_j \sim \mathcal{N}(0, \Gamma)$. A graphical representation of the resulting trajectory and the simulated observations is given in Figure 5.8. In particular, the membrane potential time course (black dashed line) and the resulting dataset (grey dots) are presented in Figure 5.8a, while

³The rheobase of a neuron is the minimal value of input current which elicits an action potential.

⁴Here $J = 12000$. Also, the entries of the vector variable x include all somatic state variables such as $V^{(\text{soma})}$, $m_{\text{Nat2}}^{(\text{soma})}$, etc.

the corresponding trajectories of some relevant state variables of the somatic compartment are shown in Figure 5.8b. Note that the latter are plotted in the transformed scale (log-scale for variable $[Ca^{2+}]_{in}$ and logit-scale for the remaining variables) according to the discussion about bounded variable presented in Section 4.3.

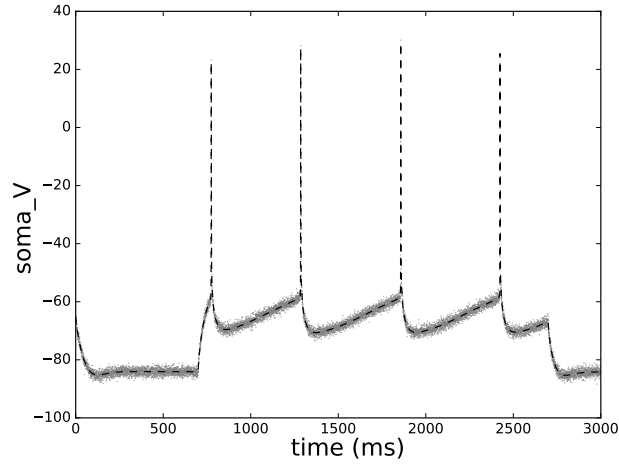
In order to test the performance of the data assimilation algorithms (see Section 4.4.1), we also generate some further datasets. First, the external input $I_{ext}^{(in)}$ is presented again but with a different initial condition: the end point of the preceding true trajectory x_J^\dagger is set to be the initial point of the new in-sample trajectory. We take such precaution because filter algorithms do not provide reliable estimate for hidden variables components of the initial condition. We denote this trace $x_{J:2J}^{\dagger(in)}$ and name it **in-sample** trajectory because it is the response to the same stimulus $I_{ext}^{(in)}(t)$ used to generate the true signal $x_{0:J}^\dagger$. In addition, taking $x(t_0) = x_J^\dagger$ as initial condition means that the resulting in-sample trajectory $x_{J:2J}^{\dagger(in)}$ is a continuation of the true signal trajectory $x_{0:J}^\dagger$.

The in-sample trajectory $x_{J:2J}^{\dagger(in)}$ is pictured in Figure 5.9a (note the different initial condition with respect to Figure 5.8a). In order to provide a meaningful measure of performance of the DA methods employed in Chapter 9, we also consider the corresponding dataset $y_{J:2J}^{(in)}$ resulting from the usual additive Gaussian noise (grey dots). Let us stress the fact that these last observations are not used in the data assimilation process, but only serve as a yardstick to compute the performance score (4.13) defined in Section 4.4.1.

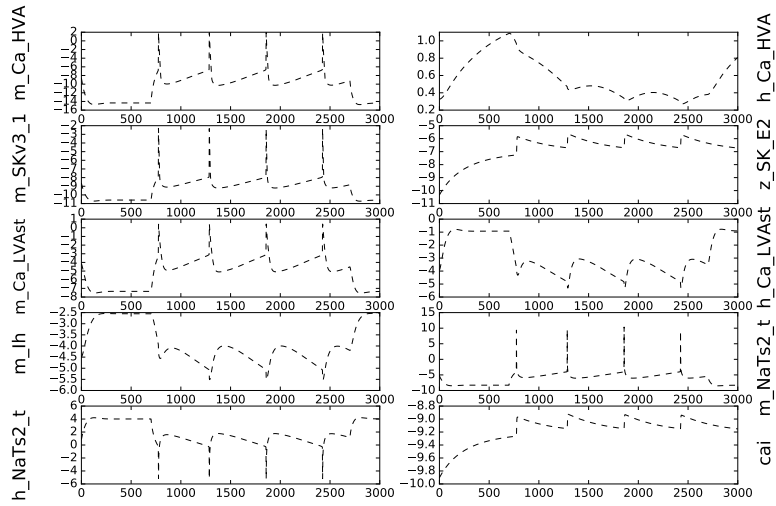
Using the same initial condition $x(t_0) = x_J^\dagger$, we apply inputs $I_{ext}^{(out1)}(t)$ and $I_{ext}^{(out2)}(t)$ in order to generate two distinct **out-of-sample** trajectories: $x_{J:2J}^{\dagger(out1)}$ (whose V -component is pictured in Figure 5.9b) and $x_{J:2J}^{\dagger(out2)}$ (Figure 5.9c). The corresponding observations are denoted $y_{J:2J}^{(out1)}$ and $y_{J:2J}^{(out2)}$, respectively.

5.4 Mathematical properties of single-neuron models

The concluding section of Chapter 5 contains an overview of the main results concerning the mathematical properties of single-neuron models. Since neuron models are mostly given in form of system of ODEs here, we mainly focus on the properties of this type of models. In particular, in Section 5.4.1 mention some basic well-posedness results for Hodgkin-Huxley-type models, and Section 5.4.2 in a brief discussion of bifurcation theory applied to mathematical models of single neurons is given, as well as some numerical aspects which are

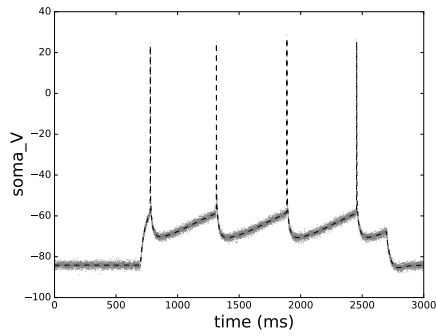


(a) True trajectory for the somatic membrane potential (black dashed line) and the simulated observations (grey dots) obtained by adding a Gaussian noise with a standard deviation of 1 mV.

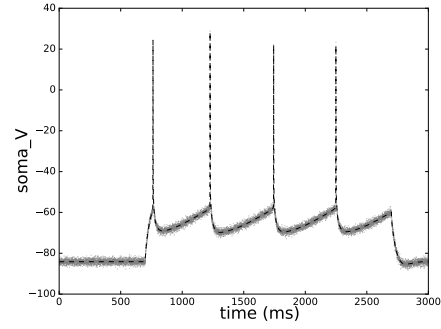


(b) Somatic state variables. From left to right, from top to bottom: $\text{logit}(m_{\text{CaHVA}})$, $\text{logit}(h_{\text{CaHVA}})$, $\text{logit}(m_{\text{Kv3.1}})$, $\text{logit}(z_{\text{SK}})$, $\text{logit}(m_{\text{CaHVA}})$, $\text{logit}(m_{\text{CaLVA}})$, $\text{logit}(h_{\text{CaLVA}})$, $\text{logit}(m_{\text{H}})$, and $\text{logit}(m_{\text{Nat2}})$, $\text{logit}(h_{\text{Nat2}})$, and $\log([Ca^{2+}]_{\text{in}})$.

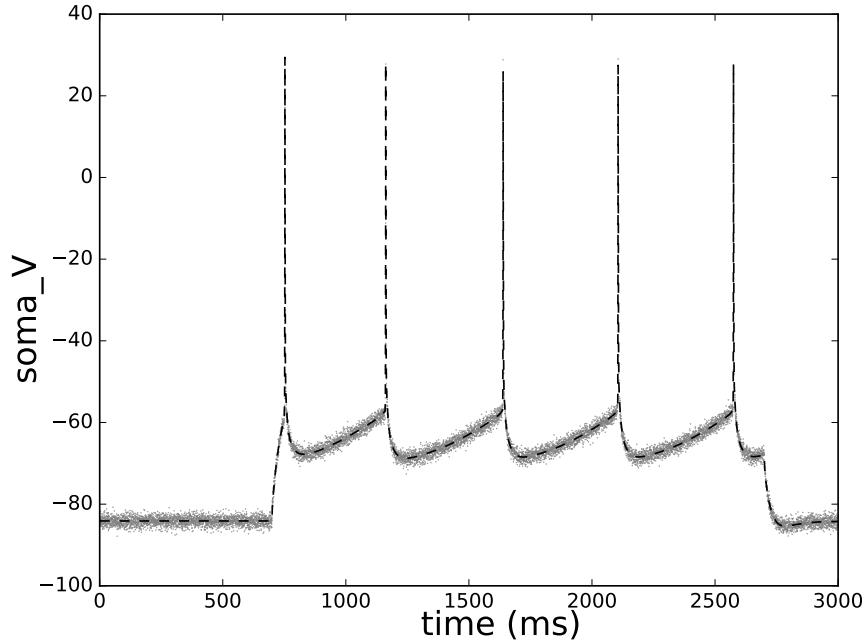
Figure 5.8: Plot of the simulated dataset $y_{0:J}$ (grey dots in (a)) and the underlying true trajectory $x_{0:J}^\dagger$ (black dashed lines) in the data assimilation time window [0ms, 3000ms] with input current $I_{\text{ext}}^{(\text{in})}(t)$.



(a) $x_{J:2J}^{\dagger(\text{in})}$ (back dashed line) and $y_{J:2J}^{(\text{in})}$ (grey dots)



(b) $x_{J:2J}^{\dagger(\text{out1})}$ (back dashed line) and $y_{J:2J}^{(\text{out1})}$ (grey dots).



(c) $x_{J:2J}^{\dagger(\text{out2})}$ (back dashed line) and $y_{J:2J}^{(\text{out2})}$ (grey dots).

Figure 5.9: Somatic potential traces and corresponding observations for performance evaluation of DA methods. The initial membrane potential is approximately $V = -85$ mV for all traces. The external input current is $I_{\text{ext}}^{(\text{in})}(t)$ for (a), $I_{\text{ext}}^{(\text{out1})}(t)$ for (b), and $I_{\text{ext}}^{(\text{out2})}(t)$ for (c). Note that increasing I_{dep} results in an increasing number of action potentials (four action potentials in the in-sample time window and five in the second out-of-sample time window).

important when simulating neuron models. Finally, in Section 5.4.3, we return to discussing well-posedness in less standard cases.

5.4.1 Well-posedness of ODE single-neuron models with constant input current

Let us start off by considering the autonomous version of the toy model (5.8) introduced in Section 5.2. The fact that the ODEs system is autonomous means, in such neuron model, that the input-current term is constant ($I_{\text{ext}}(t) \equiv I_{\text{ext}}$), and the neuron's dynamics is completely determined by the value of the applied current I_{ext} , the intrinsic ionic currents I_K , I_{Na} e I_L , and the initial condition

$$\left\{ \begin{array}{l} \dot{V} = \frac{1}{C} \left[I_{\text{ext}} - \bar{g}_K a(V - E_K) \right. \\ \quad \left. - \bar{g}_{\text{Na}} b_{\infty}(V)(V - E_{\text{Na}}) - \bar{g}_L(V - E_L) \right] \\ \dot{a} = (a_{\infty}(V) - a)/\tau_a, \\ V(0) = V_0 \\ a(0) = a_0 \end{array} \right., \quad t \in [0, T_f]. \quad (5.16)$$

We remark that it is possible to establish the well-posedness of such Cauchy problem by invoking the classical Existence and uniqueness theorem (for the precise statement of such theorem see, for instance, [35, Theorem 10.14]). Indeed, since both $a_{\infty}(V)$ and $b_{\infty}(V)$ are $C^{\infty}(\mathbb{R})$ functions (recall their sigmoid-shaped definition given in (5.9)), the autonomous vector-field (r.h.s. of the first two equations in (5.16)) is (at least) continuously differentiable. This argument alone is enough to guarantee the local Lipschitz continuity of the vector field, so that the Existence and uniqueness theorem can be applied and the Cauchy problem is well-posed.

In addition, it is also very simple to show that any solution having a biologically-meaningful initial condition is bounded, at least in the case where $I_{\text{ext}} = 0$ [46]: since $a = 0 \Rightarrow \dot{a} > 0$ and $a = 1 \Rightarrow \dot{a} < 0$, it follows that for all t we have $a(t) \in [0, 1]$ whenever the initial condition is in the same interval ($a_0 \in [0, 1]$); in addition, defining $\bar{V} = \max\{E_K, E_{\text{Na}}, E_L\}$ and $\underline{V} = \min\{E_K, E_{\text{Na}}, E_L\}$, we have that $V > \bar{V} \Rightarrow \dot{V} < 0$, and $V < \underline{V} \Rightarrow \dot{V} > 0$. As a consequence, if $r > 0$, any solution of (5.16) with initial condition (V_0, a_0) in $\mathcal{Z} = [\underline{V} - r, \bar{V} + r] \times [0, 1]$ always lives in such a \mathcal{Z} .

Note that the considerations made here about the toy model hold for any single-compartment conductance-based neuron model which includes any of the deterministic ionic currents mentioned in Section 5.1.1 (even the calcium, and calcium-dependent ionic currents).

5.4.2 Qualitative analysis and numerical aspects in neuronal modelling

Now that we briefly discussed the existence and uniqueness of the solution for the autonomous ODEs dynamical system (5.16), we can ask ourselves: how does such neuron model behave qualitatively speaking? Does the system's solutions endlessly oscillate producing a continuous train of action potentials or they eventually converge to some attracting point in the state space?

As exemplified in Figure 5.10 (blue trace), when no input current is present ($I_{\text{ext}} = 0 \mu\text{A cm}^{-2}$) the membrane potential quickly reaches the neuron's resting potential $V_{\text{rest}} \approx -60 \text{ mV}$. In the language of dynamical systems, one says that the model's solutions converge to a **stable equilibrium point**⁵. For simple reduced-order models, it is possible to explicitly compute the equilibrium point(s) by intersecting the nullclines $\{(V, a)^T \mid \dot{a} = 0\}$ and $\{(V, a)^T \mid \dot{V} = 0\}$, the most notable example being the FitzHugh-Nagumo model [72, 162]. In addition, when an equilibrium point is available in closed form, it is possible to evaluate its stability by checking the spectrum of the vector field's Jacobian evaluated at the equilibrium. However, such exact calculations are not possible for more complex models such as the Hodgkin-Huxley model.

If the input current intensity is increased, the attracting equilibrium do not appear to be present as the system's solutions are attracted by a limit cycle which corresponds, in electro-physiological terms, to a tonic spiking activity of the neuron. In addition, further increasing the input current results in a larger limit cycle amplitude, as shown by the orange and green trace in Figure 5.10. Note that limit cycle are hardly tractable by analytical methods. Nevertheless, for any neuron model it is possible to perform qualitative analysis through numerical bifurcation theory. To this end, the MatCont continuation toolbox [81] can be used as well as other software (e.g. XXPaut [65]).

What occurs in model (5.16) is that for $I_{\text{ext}}^* = 14.66 \mu\text{A cm}^{-2}$ the stable equilibrium undergoes a **supercritical Andronov-Hopf bifurcation** (first Lyapunov coefficient $l_1 = -1.19\text{e-}2$): for $I_{\text{ext}} < I_{\text{ext}}^*$ the system has a unique globally attracting equilibrium; such equilibrium becomes unstable for $I_{\text{ext}} \geq I_{\text{ext}}^*$ when a stable limit cycle appears. The corresponding bifurcation diagram is pictured in Figure 5.11, where the stable equilibrium present for $I_{\text{ext}} = 0 \mu\text{A cm}^{-2}$ is marked by a blue cross. Varying I_{ext} , a globally attracting

⁵In what follows, the knowledge of several notions of dynamical systems and bifurcation theory is required. We omit them here for the sake of brevity, but we refer to Chapter 1 [131] (or any other textbook dealing with ODEs systems) for all the necessary definitions concerning dynamical systems (e.g. stable/unstable/asymptotically stable equilibrium point/limit cycle) and to Chapter 2, 3, 7 of the same book for an exhaustive presentation of one-parameter bifurcations.

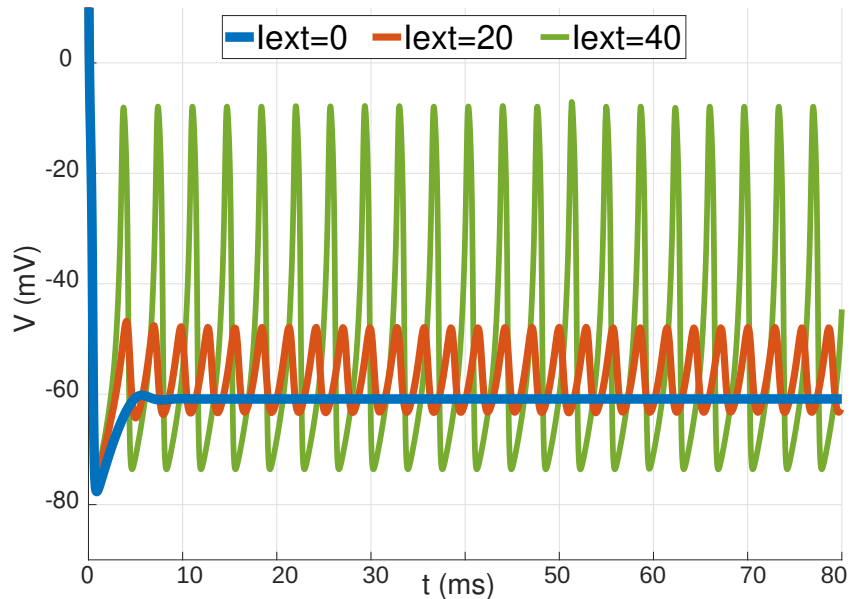


Figure 5.10: Sample solution of model (5.16) for different values of constant input current I_{ext} all with the same initial condition $(V_0, a_0)^T = (60 \text{ mV}, 0.5)^T$.

equilibrium point continue to exist (plain cyan line) until it undergoes the Hopf bifurcation (red dot) and becomes unstable (dotted cyan line). Then, a stable limit cycle whose size in the (V, a) plane increases with I_{ext} appears (plain black lines). The attracting limit cycles present for $I_{\text{ext}} = 20, 40 \mu\text{A cm}^{-2}$ pictured in Figure 5.10 are highlighted in Figure 5.11 with the corresponding orange and green color. Such bifurcation diagram allows one to completely characterize the qualitative behaviour of the solutions of neuronal model (5.16).

Although much more complicated bifurcation diagram can arise in other single-neuron model (typical examples include bistable systems and different bifurcations such as saddle-node and saddle-node on invariant cycle bifurcations, not to mention bifurcations with higher codimension), it is worth mentioning that bifurcation theory is one essential tool for the mathematical analysis of neuronal models. In fact, other than characterizing the activity of the modelled neurons, the qualitative analysis resulting from bifurcation-theory concepts directly translates into electrophysiologically relevant notions such as excitable systems, Hodgkin's Class I/Class II IF diagrams, integrator/resonator neurons, threshold, subthreshold oscillations, post-inhibitory action potentials etc. We skip this matters because their discussion goes far beyond the scope of this work, but we refer to [99] for a more comprehensive presentation of dynamical system theory applied to single-neuron models.

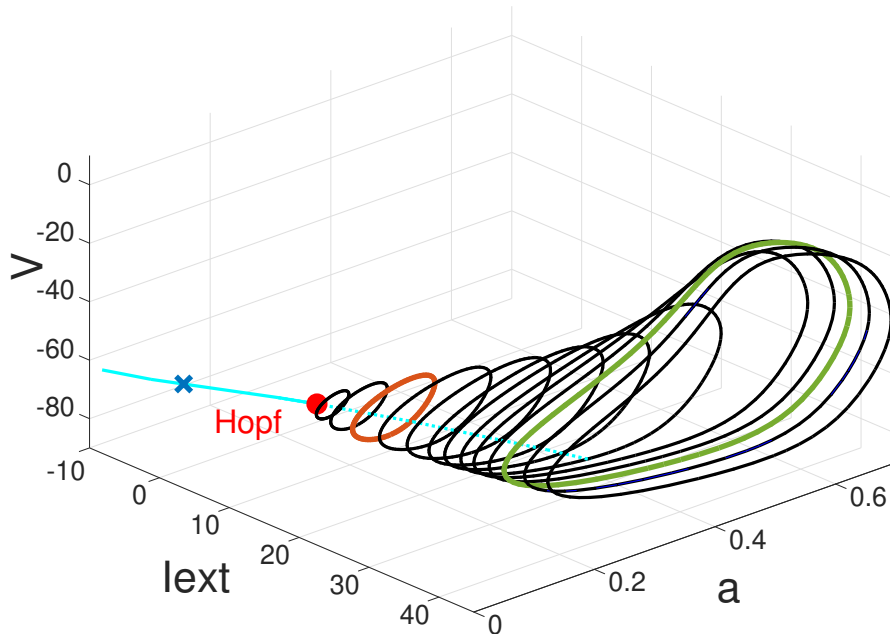


Figure 5.11: Bifurcation diagram of the two-dimensional model (5.16) with respect to parameter $I_{\text{ext}} \in [-10 \mu\text{A cm}^{-2}, 45 \mu\text{A cm}^{-2}]$.

Note that, except for a few trivial cases, conductance-based single-neuron models are not explicitly solvable, and numerically approximated solutions are usually the only available possibility. As a consequence, numerical stability and approximation issues can arise, and one needs to be aware of these not to invalidate the simulation's outcomes. Notably, the membrane potential and the activation variables of Hodgkin-Huxley-type neuron models often exhibit very different time scales, causing the resulting ODEs system to be stiff. Then, stiff solvers should be used to solve them. In order to use an accurate fixed-step solver, in our simulations of the toy model we employ a fourth-order Runge-Kutta method with small time step ($\Delta t = 0.01$ ms). On the other hand, as far as the multi-compartment BBP model is concerned, several other numerical aspects need to be taken into account. In fact, its simulation requires to discretize both in time and space of a unidimensional cable PDE (see page 116). However, most of these issues are already accounted for by the NEURON simulation environment implementation, and we refer to [37, Chapter 4] and reference therein for a complete overview of this matter. Let us just report the matter of the numerical stability of the multi-compartmental model solver: to

guarantee unconditional stability, the NEURON simulation environment only allows the backward-Euler method and the Crank-Nicolson method as fixed-step solvers. Note however, that although the latter is more accurate ($o(\Delta t^2)$ versus $o(\Delta t)$) it is also oscillation-prone (see also [86, Section 3.3.5]) so that we opted for the backward-Euler solver in our simulations.

5.4.3 More on well-posedness of single-neuron models

In this section, we briefly return on the well-posedness of single-neuron models. In fact, in the previous sections we only mentioned neuron models with constant input current $I_{\text{ext}}(t) \equiv I_{\text{ext}}$. Actually, classical Existence and uniqueness theorem in principle apply to time-dependent input currents too, as long as $t \mapsto I_{\text{ext}}(t)$ is a continuous function. Unfortunately, the models we presented in this chapter⁶ and which are used for data-assimilation experiments in Part III all involve an input current $I_{\text{ext}}(t)$ which is discontinuous in time (see Section 5.2.1 and Section 5.3.3). For such reason, in this section we present some results concerning the existence and uniqueness of the solution of ordinary differential equations systems of the form

$$\begin{cases} \dot{z} = F(t, z) \\ z(t_0) = z_0, \end{cases} \quad z \in \mathcal{Z} \subset \mathbb{R}^{d_z}, \quad t \in [t_0, T_f]; \quad (5.17)$$

where the vector field F is discontinuous in the time variable t . Then, in order to give a more comprehensive overview of the mathematical properties of existing single-neuron models, we also mention some results concerning neural models given in partial differential equation (PDE) form.

ODEs systems with time-discontinuous vector field

To begin with, consider that the following **Carathéodory conditions** are the most widely known hypotheses which ensure the existence of a solution for an ODE system with time-discontinuous vector field:

- i)* the function $z \rightarrow F(t, z)$ is continuous for almost all $t \in [0, T_f]$;
- ii)* the function $t \rightarrow F(t, z)$ is measurable for all $z \in \mathcal{Z}$;
- iii)* it exists an integrable scalar function $m(t)$ (i.e., $\exists m \in L^1([t_0, T_f])$) such that $|F(t, z)| \leq m(t)$ for all $z \in \mathcal{Z}$.

⁶Namely, the Morris-Lecar-like single-compartment single-neuron model (5.8) and the BBP multi-compartment single-neuron model L23_PC_cADpyr229.1 identified by equations (5.11), (5.12), (5.13), and (5.14).

Under these three conditions, the **Carathéodory existence theorem** [71, Theorem 1] guarantees the existence of an extended⁷ solution of the initial value problem (5.17).

Example 5.3 (Existence of the solution for the neuron toy model).

As an example, consider a function of the time variable t and the state variable $z = (z_1, z_2)^T$ defined by

$$\begin{aligned} F(t, z) &= \begin{pmatrix} F_1(t, z) \\ F_2(t, z) \end{pmatrix} \\ &= \begin{pmatrix} \frac{1}{C} \left[-\bar{g}_K z_2 (z_1 - E_K) - \bar{g}_{Na} f^{(c)}(z_1) (z_1 - E_{Na}) - \bar{g}_L (z_1 - E_L) + I_{\text{ext}}(t) \right] \\ (f^{(c)}(z_1) - z_2) / \tau_a \end{pmatrix}, \end{aligned}$$

where $f^{(c)}$ is given by (5.3c). If we rename $z_1 = V$ and $z_2 = a$, with such definition the ODE system $\dot{z} = F(t, z)$ corresponds to the toy model for a single neuron with time-dependent input current (5.8). In practice, we assume that the state variable z lies in a bounded state space of the form $\mathcal{Z} = [V_{\min}, V_{\max}] \times [0, 1]$, and that the input current $I_{\text{ext}}(t)$ is a piecewise constant function as defined in (5.10) with parameter values fixed as those sampled in Section 5.2.1.

Then, it is elementary to verify that the vector field F satisfies the a.e.-continuity and measurability Carathéodory conditions *i*) and *ii*). Moreover, defining

$$\begin{aligned} m(t) &:= \frac{1}{C} \left[\bar{g}_K \max_{z \in \mathcal{Z}} |z_1 - E_K| + \bar{g}_{Na} \max_{z \in \mathcal{Z}} |z_1 - E_{Na}| + \right. \\ &\quad \left. + \bar{g}_L \max_{z \in \mathcal{Z}} |z_1 - E_L| + |I_{\text{ext}}(t)| \right] + \frac{2}{\tau_a} \end{aligned}$$

the third Carathéodory condition *iii*) is fulfilled as well, guaranteeing the existence of at least one solution of neuron model (5.8) even when the input current $I_{\text{ext}}(t)$ is discontinuous. ♣

In order to ensure uniqueness of such solution, a further condition is needed [71, Theorem 2]:

- iv*) it exists an integrable scalar function $k(t)$ (i.e., $\exists k \in L^1([t_0, T_f])$) such that for all t and for all $z_1, z_2 \in \mathcal{Z}$, the vector fields F satisfies

$$|F(t, z_1) - F(t, z_2)| \leq k(t) |z_1 - z_2|,$$

⁷Informally speaking, an extended solution of (5.17) is a local, absolutely continuous, almost everywhere differentiable function which satisfies a.e. the ODE system conditions. See [15, pag 135] and definitions therein for a more precise definition of **extended solution** for discontinuous ODE systems.

which is a sort of time-dependent Lipschitz condition, related to Giuliano's uniqueness theorem⁸ [15, Theorem 3.5.1].

Note that, for the moment, we completely disregarded vector fields which are also discontinuous in the space variable. An example of such models in the neurobiology context is the deterministic leaky integrate-and-fire neuron model [33], whose stochastic counterpart is briefly described in the following chapter. We redirect the interested reader to [71, Chapter 2], which presents an overview of classical sufficient conditions for the well-posedness of ODE initial value problems with vector field which is discontinuous in space.

PDE single-neuron models

Even though in this work we treat the realistic BBP model described in Section 5.3 as a ODEs system, it is worth mentioning that multi-compartment single-neuron models originate from a reaction-diffusion PDEs system of the form of infinite unidimensional cable equations

$$C\partial_t V = \frac{1}{r}\partial_{xx}V + f \quad (5.18)$$

where $V(t, x)$ is the membrane potential of the neuron at position $x \in \mathbb{R}$ at time t , and r is a diffusion coefficient proportional to the axial resistivity R (cfr. (5.1)). The non-linear function f is what characterizes the neuron model, and for conductance-based models it is of form $f(V, \{a_{\text{ion}}\}_{\text{ion}}) = -\sum_{\text{ion}} I_{\text{ion}}(V, a_{\text{ion}})$, where the dynamics of the (in)activation variable is governed by

$$\partial_t a_{\text{ion}} = c_{a_{\text{ion}}}\partial_{xx}a_{\text{ion}} + (a_{\text{ion},\infty}(V) - a_{\text{ion}})/\tau_{a_{\text{ion}}}(V) \quad (5.19)$$

with $c_{a_{\text{ion}}} \geq 0$ being the corresponding diffusion coefficient. To give an example, in the original Hodgkin-Huxley model $a_{\text{ion}} = m, n, h$. However, for other reduced-order neuron models f may show a different form (e.g., in the FitzHugh-Nagumo model, f is a cubic polynomial in the V variable).

For the sake of compactness, we henceforth adopt the notation $\mathbf{V}(t, x)$ to denote the vector-valued unknown of the PDEs system (5.18)-(5.19) independently of the precise form they assume (i.e. $\mathbf{V}(t, x) = (V(t, x), \dots)^T$ including

⁸Giuliano's uniqueness theorem states that, if $F(t, z)$ satisfies the Carathéodory conditions *i)-iii)* in $(t_0, t_0 + a) \times \{|z|_2 < +\infty\}$, and for $(t, z), (t, z') \in (t_0, t_0 + a) \times \{|z|_2 < +\infty\}$

$$(F(t, z) - F(t, z')) \cdot (z - z') \leq k(t)g(|z - z'|_2^2)$$

for some $k(t) \geq 0$ integrable (according to Lebesgue's definition) in $[t'_0, t'_1]$ for all $t_0 < t'_0 < t'_1 < t_0 + a$, and some continuous function $g(x) > 0$ defined for $x > 0$ and such that $g(0) = 0$ and $\lim_{\epsilon \rightarrow 0^+} \int_{\epsilon}^{\epsilon+a'} \frac{dx}{g(x)} = +\infty$, then the ODE system (5.17) has at most one extended solution in $(t_0, t_0 + a)$. Taking $g(x) = x$ results in the above condition *iv)*.

all (in)activation variables considered in the model). For unbounded domains, the PDEs system is normally coupled to the initial condition $\mathbf{V}(0, x) = \mathbf{V}_0(x)$ whereas boundary conditions are needed too if the spatial domain is a bounded interval. In what follows, we mention two type of results concerning the solution's existence and uniqueness that can be applied to this type of models.

First, as far as well-posedness is concerned, in [43] the existence and unicity for all $t > 0$ of the solution of the PDE four-dimensional Hodgkin-Huxley model is proved for bounded and uniformly continuous initial conditions that tend to zero⁹ for $|x| \rightarrow 0$ (see Example 1 at page 208). The proof is based on the existence of invariant rectangles in the state space which are mapped into themselves by the differential operator defined by (5.18)-(5.19). Note that the same argument holds for the PDE version of the two-dimensional FitzHugh-Nagumo model (see page 209 of the same reference). Consult [192, Chapter 14] for a more extensive discussion of the invariant-region method.

Then, what one usually looks for are solutions of the PDE problem in form of travelling waves. By **travelling wave** we mean a solution of (5.18) such that

$$V(t, x) = U(x + ct),$$

where, defining $\xi = x + ct$, $U(\xi)$ is called the **shape of the travelling wave**, and $c \in \mathbb{R}$ is the (constant) **speed of the travelling wave**. In 1977, G. Carpenter proved the existence of such type of solutions for the Hodgkin-Huxley PDE model [38]. Note that travelling wave solutions are extremely important in neuronal modelling as they describe the propagation of action potentials in the axon and the dendritic tree.

Applying the above-mentioned change of variable to (5.18) leads to a ODE problem that has U as unknown, and proving the existence (and unicity) of a c^* that guarantees the existence of a physiologically meaningful U is one of the main aims of the analytical study of (5.18). For neuron models for which $f(V)$ is cubic-shaped, it is possible to perform analytic considerations to prove the existence and unicity of c^* through **shooting arguments** and, in some specific cases even exactly compute it (see [113, Section 6.2.1, Section 6.3.1]). However, the shooting argument can only be applied numerically in the Hodgkin-Huxley model, as the authors did in their original paper to estimate the propagation speed of action potential across the squid axon. To conclude, consider that another useful reference to delve into the existence, uniqueness and stability of travelling wave solutions in unidimensional cable-equation-like PDE models is [66, Chapter 6].

⁹i.e., $\mathbf{V}_0(x)$ such that $\lim_{|x| \rightarrow 0} \mathbf{V}_0(x) = \mathbf{0}$.

Chapter 6

Neural network models

In this chapter, we address the concept of biological neural networks. In particular, in Section 6.2 we present a network model of leaky integrate-and-fire (LIF) neurons which is then used in Part III to test a novel methodology for parameter estimation in systems that involve a large number of parameters.

But first, in Section 6.1 we give a very brief overview of the composition of the neocortical microcircuit studied by the BBP and document the reason behind the choice of model L23_PC_cADpyr229_1 in Section 5.3 as an instance of neuron model from the BBP neural network.

6.1 Neocortical microcircuit composition

The *in silico* reconstruction presented in [149] represents a 0.29 mm³ portion of Winstar rat neocortex constituted of about 31 000 neuron models. The neuron models are subdivided in six layers and the quantity of neurons per layer is the following:

Layer 1	338 neurons
Layer 2 and 3	7 524 neurons
Layer 4	4 656 neurons
Layer 5	6 114 neurons
Layer 6	12 651 neurons.

All neuron models are classified first according to their **morphological type** (m-type), and then according to the electrophysiology they exhibit, i.e. their **electrical type** (e-type). There is a total of 55 m-types (see Figure 6.1), such

as the model of interneuron¹ morphology of bipolar cells (BP), of bitufted cells (BTC), chandelier cells (ChC), double bouquet cells (DBC), large basket cells (LBC), Martinotti cells (MC), nest basket cells (NBC), small basket cells (SBC), and neurogliaform cells (NGC). Also, examples of principal (excitatory) cell morphology include the one of **pyramidal cells** (PC), but also of the star pyramidal cells (SP) and the thick-tufted pyramidal cells with a late bifurcating apical tuft (TTPC1). On the other hand, the electrophysiological types are eleven in total (see Figure 6.2) and, combined with the possible morphological types, they generate a total of 208 **morpho-electrical types** (me-type), since not all combination are present in the microcircuit).

Let us now focus on layer two and three. A summary of the in-layer and in-microcircuit me-type frequency for neuron models of the second and third layer is given in Table 6.4. While the interneuron model with highest in-layer incidence is the continuous accommodating (cAC) layer 2 and 3 (L23) Martinotti cell (MC) model (3.59% of the neuron models for layer 2 and 3), the model with highest frequency is the model of continuous adaptive (cAD) pyramidal cell L23_PC_cAD (78.11% of the neuron models across layers two and three are of this me-type) which we described in detail in the previous chapter.

Now, we move to a complete model of biological neural networks whose units are modelled as leaky integrate-and-fire neurons.

6.2 Network model of leaky integrate and fire neurons

In this section we present a simple model of neural network constituted by a total of $n_E + n_I$ neurons: the first n_E neurons are excitatory cells, and the second n_I are inhibitory neurons. In what follows we assume that the excitatory neurons are labelled by the integer index $i = 1, \dots, n_E$, whereas the inhibitory neurons are labelled by $i = n_E + 1, \dots, n_E + n_I$. Simplifying the underlying biology, one can think of the neurons connectivity as an adjacency matrix $A = (a_{ij})_{i,j=1}^{n_E+n_I}$, where $a_{ij} = 1$ if neuron i and j are connected through chemical synapses, whereas $a_{ij} = 0$ if no synapse between neuron i and neuron j exists.

One can consider different models both for the neurons and for the synapses, obtaining different network models. In our case, we consider that each neuron activity is described by a **leaky integrate-and-fire** (LIF) neuron model [33, 13] (model adapted from the original 1907 proposal [133], translated in English

¹An interneuron is a inhibitory neural cell

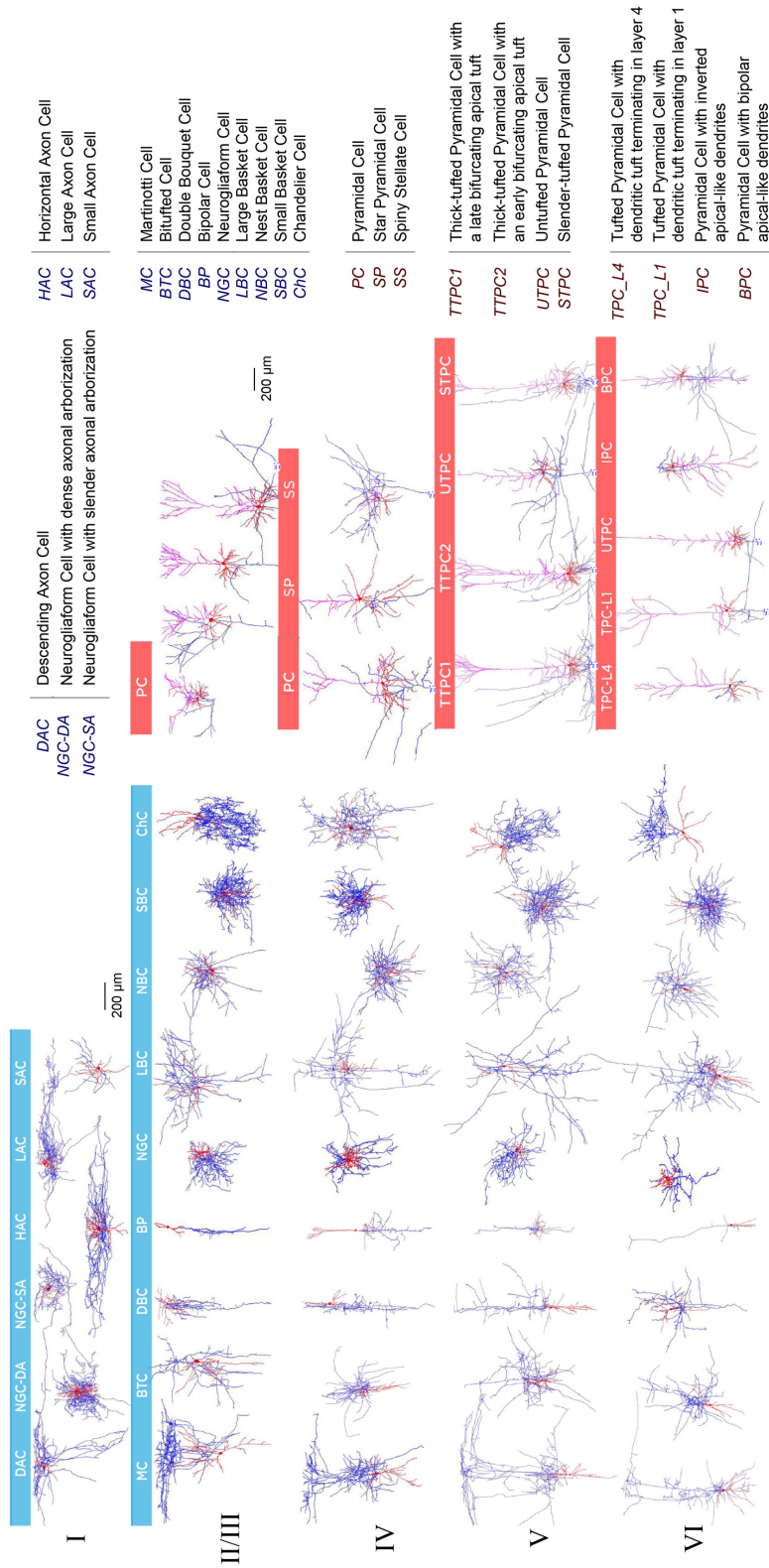


Figure 6.1: Summary of the morphological types (m-types) considered in the neocortical microcircuit. Source: [149].

Layer	m-type	e-type	#neurons	in-layer%	circuit%
L23_	_ BP	_ bAC	3	0.04%	0.01%
		_ bIR	4	0.05%	0.01%
		_ bNAC	7	0.09%	0.02%
		_ cAC	7	0.09%	0.02%
		_ cNAC	4	0.05%	0.01%
		_ dSTUT	3	0.04%	0.01%
	_ BTC	_ bAC	15	0.19%	0.05%
		_ bIR	7	0.09%	0.02%
		_ bNAC	23	0.31%	0.07%
		_ cAC	41	0.54%	0.13%
		_ cNAC	18	0.24%	0.06%
	_ ChC	_ cAC	23	0.31%	0.07%
		_ cNAC	23	0.31%	0.07%
		_ dNAC	15	0.19%	0.05%
	_ DBC	_ bAC	12	0.16%	0.04%
		_ bIR	32	0.43%	0.10%
		_ bNAC	70	0.93%	0.22%
		_ cAC	61	0.81%	0.19%
	_ LBC	_ bAC	35	0.74%	0.11%
		_ bNAC	27	0.36%	0.09%
		_ cAC	108	1.44%	0.34%
		_ cNAC	74	0.98%	0.24%
		_ cSTUT	22	0.29%	0.07%
		_ dNAC	188	2.50%	0.60%
	_ MC	_ bAC	10	0.13%	0.03%
		_ bNAC	10	0.13%	0.03%
		_ cAC	270	3.59%	0.86%
		_ cNAC	33	0.44%	0.11%
		_ dNAC	10	0.13%	0.03%
	_ NBC	_ bAC	14	0.19%	0.04%
		_ bNAC	6	0.08%	0.02%
		_ cAC	65	0.86%	0.21%
		_ cIR	6	0.08%	0.02%
		_ cNAC	80	1.06%	0.26%
		_ dNAC	97	1.29%	0.31%
	_ NGC	_ bNAC	5	0.07%	0.02%
		_ cAC	5	0.07%	0.02%
		_ cNAC	41	0.54%	0.13%
		_ cSTUT	5	0.07%	0.02%
	_ PC	_ cAD	5 877	78.11%	18.75%
	_ SBC	_ bNAC	60	0.80%	0.19%
		_ cAC	60	0.80%	0.19%
		_ dNAC	46	0.61%	0.15%

Table 6.4: Frequency of layer 2 and 3 me-types in both the in-layer and in-microcircuit composition. The blue-highlighted L23_MC_cAC model is the interneuron model with highest in-layer incidence, whereas the red-highlighted L23_PC_cAD model is the model with highest in-microcircuit incidence.

in [29]), which is a popular choice for simulating the large neural network dynamics while including a microscopic detail [49, 30].

In particular, we consider LIF models in stochastic differential equation (SDE) form, with resting potential V_{rest} , leakage constant L_i , and incoming input given by the sum of a constant applied current I_{ext} and synaptic inputs from both inhibitory and excitatory neurons. In formulae this writes,

$$dV_i = \left(-\frac{V_i - V_{\text{rest}}}{L_i} - I_{\text{Esyn}} - I_{\text{Isyn}} + I_{\text{ext}} \right) dt + \sigma_V dW, \quad (6.1)$$

for $i = 1, \dots, n_E + n_I$, where W denotes a standard Weiner process (also called white Brownian motion). The above equation describes the membrane potential dynamic in the sub-threshold regime. On the other hand, in a LIF model when the membrane potential V_i reaches the threshold V_{th} , the i -th neuron emits a spike and its potential is reset to the resting value V_{rest} . Furthermore, we set a lower bound V_{low} such that, if the membrane potential V_i goes below such value $V_i < V_{\text{low}}$, then it is automatically restored to the lower bound, $V_i = V_{\text{low}}$. For a thorough discussion of both the concept of solution of an SDE and the numerical methods that can be employed to produce approximate solutions, we refer to [117]. Consult [183] for a synthetic tutorial on the same topic.

Now, to complete the description of the LIF model (6.1), we need to specify the form of the synaptic currents I_{Esyn} and I_{Isyn} .

6.2.1 Synaptic current modelling

Both excitatory and inhibitory synaptic currents are assumed to be ohmic, i.e. $I_{\text{syn}}(V, g) = g(V - E_{\text{syn}})$, with a simple model for the synaptic conductance g which involves an instantaneous jump of amplitude \bar{g}_{syn} (the maximal synaptic conductance) in each moment t_{pre} when a pre-synaptic spike is evoked, followed by an exponential decay with rate τ_{syn}

$$g(t) = \begin{cases} \bar{g}_{\text{syn}} \exp\left(-\frac{t-t_{\text{pre}}}{\tau_{\text{syn}}}\right) & t \geq t_{\text{pre}} \\ 0 & t < t_{\text{pre}} \end{cases}.$$

In case of multiple pre-synaptic spikes, a linear post-synaptic summation is assumed, so that the synaptic conductance dynamics can be described by the following ODE equation

$$\dot{g} = -\frac{g}{\tau_{\text{syn}}} + \sum_{j=1}^{\#spikes} \bar{g}_{\text{syn},j} \delta(t = t_j). \quad (6.2)$$

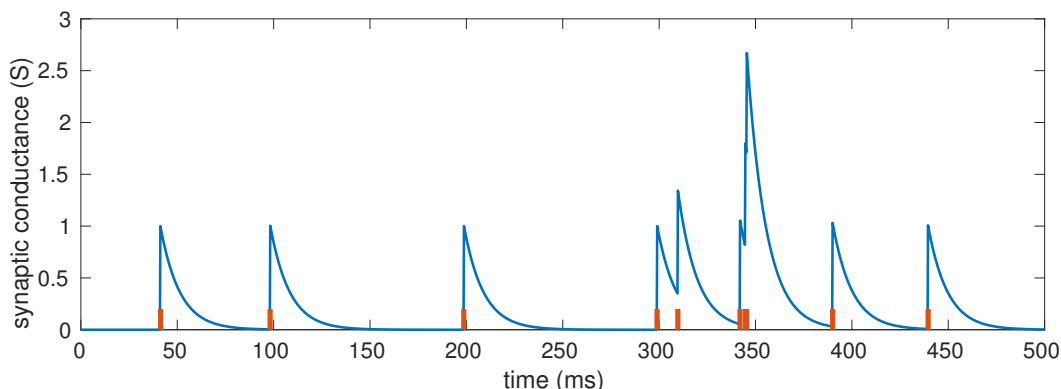


Figure 6.5: Sample time course of a single-exponential synapse $s(t)$ as given in (6.2). Parameter values are $\tau_{\text{syn}} = 10\text{ms}$ and $\bar{g}_{\text{syn}} = 1\text{S}$. The spike time sequence $\{t_j\}_{j=1}^{\#\text{spikes}}$ is such that the corresponding inter-spike intervals are independent draws from an exponential distribution of mean $\mu = 50\text{ms}$, i.e. $t_j - t_{j-1} \sim \text{Exp}(\mu)$ where $t_0 = 0\text{ms}$.

Here $\{t_j\}_{j=1}^{\#\text{spikes}}$ is the sequence of pre-synaptic spikes, $\delta(t = t_j)$ is the Dirac mass centred at $t = t_j$, and $\bar{g}_{\text{syn},j}$ is the maximal synaptic conductance relative to the j -th spike. Indeed, maximal conductances are usually different whenever the pre-synaptic source is different. Nonetheless, for the sake of simplicity in this model it is assumed that the maximal conductance for synapses of a given type only depend on the post-synaptic target, i.e. $\bar{g}_{\text{syn},j} = \bar{g}_{\text{syn}}$ for all j . Such assumption also allows the use of a fast exponential solver for equation (6.2). We refer to [145] for more details on this specific matter, and to [182] for a review of more realistic models of synaptic current. A sample time course of the synaptic ODE model (6.2) is given in Figure 6.5.

As for the original LIF neuron (6.1), the excitatory synaptic current acting on the i -th neuron is given by

$$I_{\text{Esyn}}(V_i, g_{\text{E},i}) = g_{\text{E},i}(V_i - V_{\text{th}}),$$

where the choice of V_{th} as reversal potential guarantees that $I_{\text{Esyn}}(V_i, g_{\text{E},i})$ is always negative². The corresponding synaptic state variable dynamic is governed by

$$\dot{g}_{\text{E},i} = -\frac{g_{\text{E},i}}{\tau_{\text{Esyn},i}} + \bar{g}_{\text{Esyn},i} \sum_{j=1}^{n_E} a_{ij} \delta(V_j = V_{\text{th}}), \quad (6.3)$$

where V_j denotes the membrane potential of the j -th neuron in the network. The model for the inhibitory synaptic current is analogous to (6.3), except for

²Note that the general convention requires a minus sign in front of all synaptic currents. As a consequence $-I_{\text{Esyn}}(V_i, g_{\text{E},i}) \geq 0$ guarantees an excitatory effect.

the reversal potential which is given by V_{low} in order to enforce the correct sign to the inhibitory current.

In conclusion, the complete model for the i -th LIF neuron in the neural network is given by the system of stochastic differential equations

$$\begin{cases} dV_i = \left[-\frac{V_i - V_{\text{rest}}}{L_i} - g_{\text{E},i}(V_i - V_{\text{th}}) - g_{\text{I},i}(V_i - V_{\text{low}}) + I_{\text{ext}} \right] dt \\ \quad + \sigma_V dW \\ dg_{\text{E},i} = \left[-\frac{g_{\text{E},i}}{\tau_{\text{Esyn},i}} + \bar{g}_{\text{Esyn},i} \sum_{j=1}^{n_E} a_{ij} \delta(V_j = V_{\text{th}}) \right] dt \\ dg_{\text{I},i} = \left[-\frac{g_{\text{I},i}}{\tau_{\text{Isyn},i}} + \bar{g}_{\text{Isyn},i} \sum_{j=n_E+1}^{n_E+n_I} a_{ij} \delta(V_j = V_{\text{th}}) \right] dt \end{cases}, \quad (6.4)$$

where $i = 1, \dots, n_E + n_I$. The first equation in (6.4) is intended to hold for $V_i \in [V_{\text{low}}, V_{\text{th}})$, whereas if $V_i(t) \geq V_{\text{th}}$ then $V_i(t + dt) = V_{\text{rest}}$ and if $V_i(t) < V_{\text{low}}$ then $V_i(t + dt) = V_{\text{low}}$.

6.2.2 Network parameters

For the sake of simplicity, we assume that in model (6.4) both decay constants for inhibitory synapses $\tau_{\text{Isyn},i}$ and their maximal conductances $\bar{g}_{\text{Isyn},i}$ are equal, i.e. $\tau_{\text{Isyn},i} = \tau_{\text{Isyn}}$ and $\bar{g}_{\text{Isyn},i} = \bar{g}_{\text{Isyn}}$ for all i . In the same fashion, the leakage decay constants for inhibitory neurons are assumed to be all identical, i.e. $L_i = L_{\text{I}}$ for $i = n_E + 1, \dots, n_E + n_I$.

On the other hand, we assume that the remaining parameters are different for each neuron. However, it is assumed that the different values across the neural population are homogeneous, in the sense that they are independent draws of a given probability distribution which is parametropitimized by a single hyperparameter. In practice, we assume that the leakage constant for excitatory neurons are independent draws of an exponential distribution of parameter L_{E}^{\dagger} , i.e. $L_i \sim \text{Exp}(L_{\text{E}}^{\dagger})$, for $i = 1, \dots, n_E$. Analogously, the maximal conductances for excitatory synapses are i.i.d random variables of parameter $\bar{g}_{\text{Esyn}}^{\dagger}$, i.e. $\bar{g}_{\text{Esyn}} \sim \text{Exp}(\bar{g}_{\text{Esyn}}^{\dagger})$. In the data assimilation process we presented in Section 4.1.2 the target object to be estimated is indeed this couple of hyperparameters. Such methodology – estimating the hyperparameter which parametropitmiss the probability distribution of a very large number of parameters – is, to the best of our knowledge, new to the domain of computational neurobiology.

In the following section we describe what type of collective network dynamics can be recorded and how these can be mathematically modelled.

6.2.3 Neural population activity measure

There is a number of experimental recordings which can be used to measure the population activity of a neural network. These can be rather direct but invasive measurements, such as (multi-)electrode LFP or not invasive but indirect, such as EEG, positron emission tomography (PET), magnetic resonance imaging (MRI), blood oxygenation level dependent (BOLD) fMRI, etc.

To start off, we consider a very simple model for LFP as a quantity representing the population activity. This is

$$h_{LFP}(V_1, \dots, V_{n_E+n_I}) = \frac{1}{n_E + n_I} \sum_{i=1}^{n_E+n_I} V_i, \quad (6.5)$$

i.e. a representation of local field potential as average population membrane potential. We stress that such choice is an extremely naive proxy for LFP which is not intended to be realistic. It can be improved by considering other simple models based on LIF network (see for instance the synaptic-activity based LFP proxy used in [152]) or even considering sophisticated models based on multi-compartment single-neuron models which take spatial effects into account [139]. However, more realistic LFP data models go beyond the scope of the preliminary work we intend to carry out on the LIF network model presented in this section.

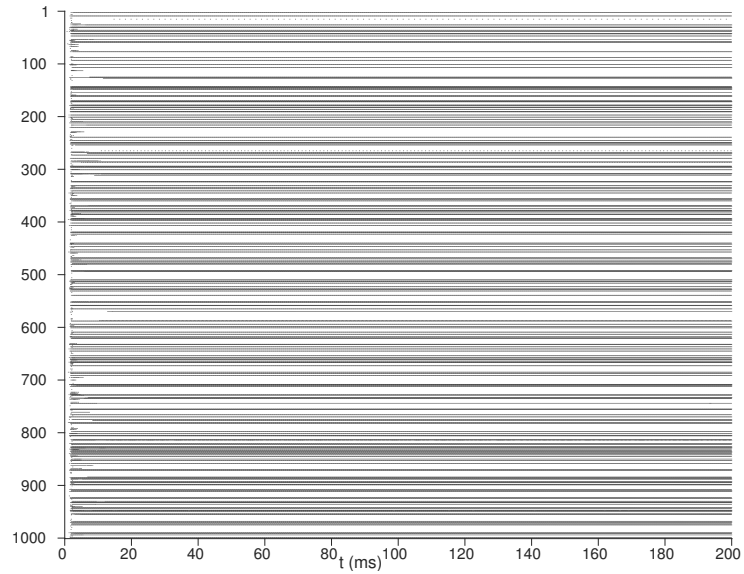
In the following chapter we illustrate the notion of spike train metrics and present some examples of such measures of spike train synchrony. Many of these distances can also be generalised to measures of neural population synchrony, so that they can be used on neural network models too. But first, we illustrate the neural network model introduced in this chapter by picturing its sample dynamics.

6.2.4 Sample network time course

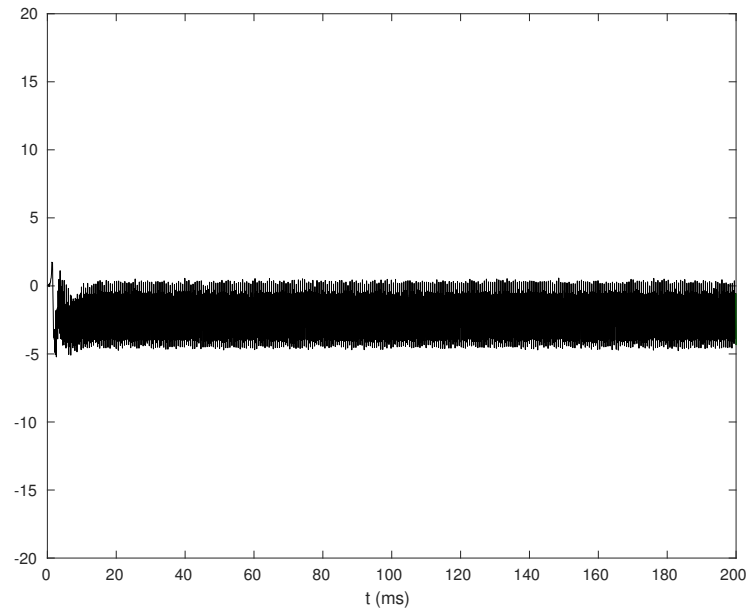
In order to illustrate the neural network model (6.4), in Figure 6.6 we plot a sample time course for a network of $n_E = 800$ excitatory neurons and $n_I = 200$ inhibitory neurons. The corresponding SDEs system is solved with the Euler-Murayama scheme with computational time step $\Delta t = 0.1$ ms. The remaining network parameters are given by $\sigma_V = 0.1$, $V_{\text{rest}} = 0$ mV, $V_{\text{th}} = 20$ mV, $V_{\text{low}} = -20$ mV, $I_{\text{ext}} = 0.1$, $L_i^\dagger = 3$, $g_{\text{Isyn}}^\dagger = 4$, $\tau_{\text{Isyn}}^\dagger = \tau_{\text{Esyn}}^\dagger = 1$. The initial condition for all neurons is distributed as a Gaussian random variable $V_i(0) \sim \mathcal{N}(0, \sigma_V)$ whereas the initial synaptic conductances are all deterministically initialised at zero.

The above panel in Figure 6.6 plots the spike times of all neurons in the network. Indeed, the y -axis lists the label i for each neuron in the network (i.e.,

i spans from 1 to $n_E + n_I = 1\,000$), whereas the horizontal axis represents time in the millisecond scale. In conclusion, the raster plot in Figure 6.6a contains a black dot with coordinates (t, i) if the i -th neuron emits a spike at time t . On the other hand, in Figure 6.6b shows the profile of the corresponding LFP proxy (6.5).



(a) Raster plot for model (6.4). The y -axis denotes the index labelling an individual neuron in the network.



(b) Mean activity (6.5) (proxy for local field potential) with Gaussian measurement noise with $\sigma_y = 0.1$ mV.

Figure 6.6: Graphical representation of a sample trajectory of model (6.4) with $n_E = 800$ excitatory neurons and $n_I = 200$ inhibitory cells. The other network parameter values which produced such trajectory are stated in Section 6.2.4.

Chapter 7

Spike train metrics

In this chapter, we introduce the concept of spike-train distance. In fact, the absolute p -metric (4.11) defined in Section 4.4.1 is a distance on the space of vector-valued discrete-time trajectories $\{0, \dots, J\} \times \mathbb{R}^\ell$ which, as a particular case, can be applied to membrane potential traces. Using this canonical mathematical notion of distance, we have a metric which compares membrane potential profiles and also quantifies possible dissimilarities in the shape of the action potentials. However, small differences in two voltage traces which potentially have little impact on the respective neurobiological properties, are prone to be excessively penalized by such non-specific metrics. Fortunately, other notions of distances exist which can be defined on the space of spike trains and which are conceived precisely to be applied in neurobiology.

Indeed, from a given membrane potential trace, the corresponding spike train can be extracted. A spike train is the sequence of time points $t = \{t_i\}$ at which the membrane potential trace exhibits a spike. By definition, such sequence disregards any information related to the action potentials shape (e.g. action potential amplitude, length of the refractory period, resting potential value etc.) and only retains the spike timing. We remark that there exist a number of spike train metrics, but describing them completely goes beyond the scope of this chapter. Nevertheless, we now present a general but sufficiently comprehensive introduction to the main groups of such metrics, and specify only some notable examples in detail.

Note that in the current section we denote a spike train A as

$$t^A = \{t_1^A, t_2^A, \dots, t_{\#A}^A\} = \{t_i^A\}_{i=1}^{\#A} \subset [0, T_f],$$

where $\#A$ denotes the number of spikes in the spike train and T_f denotes the end of the recording time window (assumed to be the same for all spike trains which are to be compared).

As synthetically reviewed in [208], we can broadly distinguish between spike-train metrics which are

- i*) based on cost-dependent transformations of one spike train into the other one (Section 7.1);
- ii*) based on embeddings of the spike trains onto a normed space (Section 7.2);
- iii*) time-adaptive distances (Section 7.3).

Note that an alternative overview of different spike metrics is available at [124]. First, let us consider the cost-based metrics.

7.1 Cost-based spike metrics

In such approach, a limited set of elementary transformations are established, and a cost is associated to each of these transformations. Then, in order to well-define the distance between two spike trains, the least expensive set of transformations needed to turn the first spike train into the second one is selected.

Among cost-based spike metrics, the family of Victor-Purpura distances [209, 210] is probably most commonly known. This includes D^{SPIKE} and D^{ISI} , which are both dependent on a scale parameter q , i.e. $D^{\text{SPIKE}} = D_q^{\text{SPIKE}}$ and $D^{\text{ISI}} = D_q^{\text{ISI}}$.

7.1.1 Victor-Purpura SPIKE distance

If we consider the first **Victor-Purpura SPIKE metric**, the set of legitimate transformations include: removing a spike (cost of deleting a single spike set to be 1), adding a spike (cost 1), and moving a single spike time of Δt (cost $q\Delta t$). See Figure 7.1 for a graphical example of how to convert a spike train t^X into another spike train t^Y only via these three transformations.

Here, the free parameter q is measured in the reciprocal of the unit of time (e.g. s^{-1}) and it represents the cost of moving a single spike of one unit of time. The larger q , the more sensitive D_q^{SPIKE} is to exact spike timing. Indeed, it can be easily shown that it is convenient to compare two spikes by shifting their spike times only when these occur within an interval of $2/q$. Otherwise, deleting and adding a new spike is preferable in such metric. On the other hand, for the limit value $q = 0$, changing spike times is free, and the corresponding metric only compares the number of spikes in the two trains.

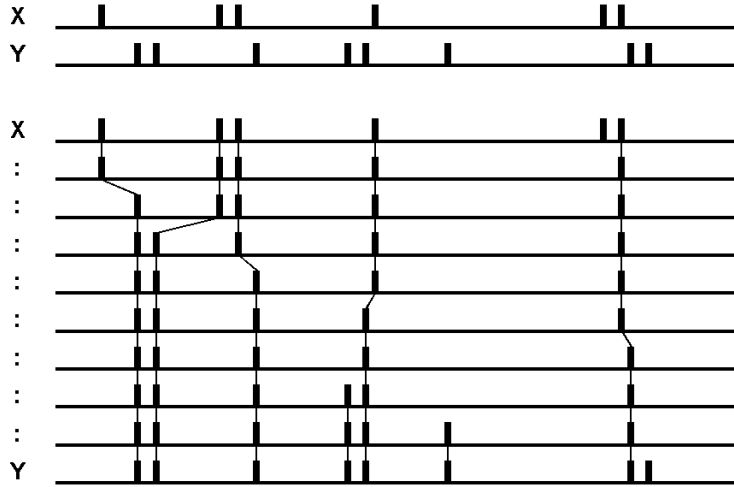


Figure 7.1: Courtesy of Thomas Kreuz. Original caption: “*Victor-Purpura spike train distance. Two spike trains X and Y and a path of basic operations (spike deletion, spike insertion, spike shift) transforming the one spike train into the other. Modified from [209]*”. Source: [124] ©

Indeed $D_{q=0}^{\text{SPIKE}}(t^A, t^B) = |\#A - \#B|$, which is an integer quantity directly proportional to the difference of spiking rates.

Note that for all spike trains t^A and t^B , and independently of q , we have that $D_q^{\text{SPIKE}}(t^A, t^B) \in [|\#A - \#B|, \#A + \#B]$. In fact, the lower bound corresponds to the case in which no time shifting or deletion of spike is needed because all spikes already match. Only the adding transformation is required to append possible extra spikes in one of the sequences. The upper bound on the other hand, corresponds to the case where no couple of spikes occur closer than $2/q$. Thus, the matching procedure can only be enforced by removing all spikes from a sequence and adding new ones in correspondence of the other train spikes.

7.1.2 Victor-Purpura ISI distance

Instead of comparing spike times, the second **Victor-Purpura inter-spike interval distance** D^{ISI} compares inter-spike intervals (ISI). In this case, the transformations allowed are the exact analogue of the SPIKE case, except that

they act on ISIs: adding or removing a inter-spike interval costs 1; shortening or extending a ISI of a Δt -wide time interval costs $q\Delta t$.

Note that, if the time resolution of the spiking activity exhibited by the spike trains is not known in advance, assessing a reasonable value of q can be rather cumbersome for both D^{SPIKE} and D^{ISI} . Many other cost-based distances can be introduced, but we refer to more specific literature for a more complete presentation (e.g. [207]).

We now turn to the second family of metrics, the one characteroptimised by an embedding of the spike train onto a normed space $(X, \|\cdot\|)$.

7.2 Embedding-based spike metrics

First, let us introduce the representation of a spike train $t^A = \{t_i^A\}_{i=1}^{\#A} \subset [0, T_f]$ as a sum of Dirac delta functions

$$\delta^A(t) := \sum_{i=1}^{\#A} \delta(t - t_i^A).$$

Such a distribution can then be projected onto a normed space using, for instance, a kernel-based transformation \mathcal{T}

$$\mathcal{T}^A(t) := (K * \delta^A)(t) = \int_{-\infty}^{\infty} K(t-s)\delta^A(s) ds. \quad (7.1)$$

In most cases, the normed space is chosen to be $L^p(\mathbb{R})$, the space of real functions endowed of the L^p norm, for some $p \geq 1$.

7.2.1 Van Rossum distance

As an example, consider the popular **van Rossum distance** which stems from taking an exponential kernel¹ $K(t) = q \exp(-qt)$ in equation (7.1) and setting the distance to be the L^2 -norm of the transformed spike trains, i.e.

$$D^{\text{Ross}}(t^A, t^B) = \left(\int_0^{\infty} [\mathcal{T}^A(t) - \mathcal{T}^B(t)]^2 dt \right)^{1/2}.$$

Note that, exactly as in the Victor-Purpura family, the parameter q sets the time scale for the comparison of the two spike trains².

¹Defined for $t \geq 0$

²We report that in the original reference [181] the inverse parameter $t_c = 1/q$ is employed.

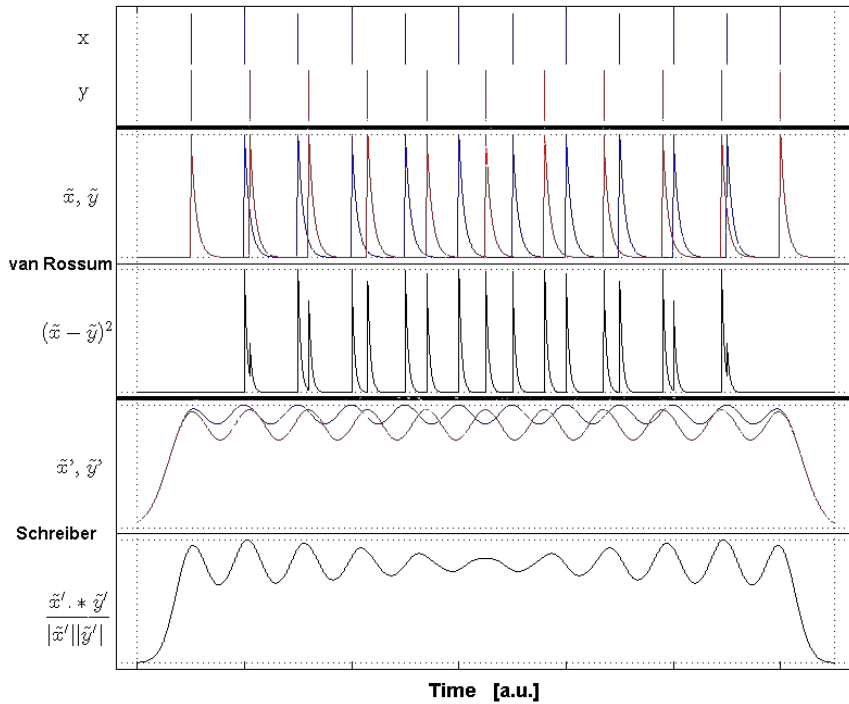


Figure 7.2: Courtesy of Thomas Kreuz. Original caption: “*Van Rossum spike train distance and Schreiber et al. similarity measure. From top to bottom: Two spike trains X and Y, exponential convolutions, Euclidian difference [181], Gaussian convolution and normaloptimisd point-wise multiplication [185]*”. Source: [124] ©

7.2.2 Similarity measures

In addition, when the projection is performed onto a normed space which is also endowed of a scalar product (which is the case for $L^2(\mathbb{R})$), the correlation coefficient

$$\rho(A, B) = \frac{\langle \mathcal{T}^A, \mathcal{T}^B \rangle}{\|\mathcal{T}^A\| \|\mathcal{T}^B\|}$$

can be considered as a similarity measure, i.e. a measure which increases when the similarity between two trains augments and attains the zero value when the dissimilarity is maximal. The measures proposed by **Haas-White** [82] and by **Schreiber et al.** [185] are two examples of such similarity measures. The former results from considering a exponential kernel in the projection operator (7.1), while the latter corresponds to a Gaussian kernel. Note that any similarity measure $\rho(\cdot, \cdot)$ can be turned into a proper metric by a composition with

the arccosine function, $D(t^A, t^B) = \cos^{-1}(\rho(A, B))$, or with the decreasing linear function $f(x) = 1 - x$, $D(t^A, t^B) = 1 - \rho(A, B)$.

As a conclusion, we observe that the convolution of a spike train with an exponential kernel approximately matches the post-synaptic potential generated by such train (cfr. (6.2)). As a consequence, both van Rossum and Haas-White distances are metrics which compare the post-synaptic effect of two spike trains. See Figure 7.2 for a graphical representation of the embedding framework in case of the exponential-kernel-based van Rossum distance, and the Gaussian-kernel-based Schreiber et al. similarity measure.

We now conclude our presentation by discussing the rather new class of time-adaptive distances.

7.3 Parameter-free spike metrics

In the last decade, a novel family of time-scale adaptive metrics has been proposed by several authors, among which Thomas Kreuz stands out for the mole and consistency of his work. Such family includes the ISI-distance [126], the SPIKE-distance [128, 129], and the SPIKE synchronization [125]. From our perspective, what is more appealing is that such metrics have the desirable property of being parameter-free. Indeed, they introduce a local firing-rate adaptation which allows one to consider multiple time scales at once.

7.3.1 ISI- and SPIKE-distance

Given a time point $t \in [0, T_f]$ and two spike trains t^A and t^B , both the ISI-distance D_I and the SPIKE-distance D_S require the definition of the spike time preceding t

$$t_P^X(t) := \max_{t_i^X \leq t} \{t_i^X\}, \quad t \in [t_1^X, t_{\#X}^X],$$

and the spike time following t

$$t_F^X(t) := \min_{t_i^X > t} \{t_i^X\}, \quad t \in [t_1^X, t_{\#X}^X],$$

as well as the local inter-spike interval

$$x_{\text{ISI}}^X(t) = t_F^X(t) - t_P^X(t).$$

The last three definitions read for $X = A, B$. To avoid ambiguity, two extra spike are inserted at $t = 0$ and $t = T_f$ to any spike train. A schematic representation of the quantities we just defined is given in Figure 7.3.

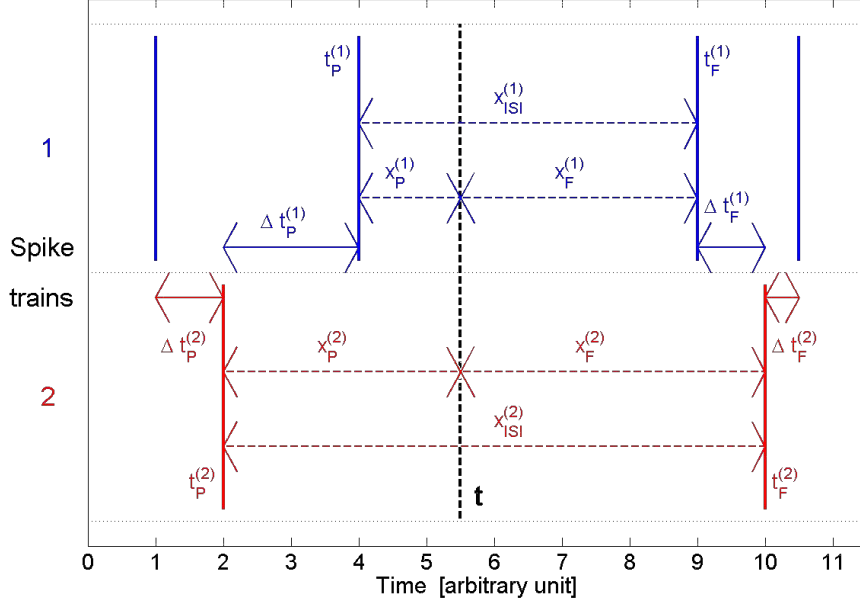


Figure 7.3: Courtesy of Thomas Kreuz. Original caption: “*Illustration of the local quantities (relative to time instant t) needed to calculate the instantaneous dissimilarity values on which the ISI- and the SPIKE-distance (and its real-time variant) are based. Modified from [129]*”. Note that in the figure, the superscripts identify the spike train, i.e. $x_{\text{ISI}}^{(1)}$ corresponds to what in the main text is referred to as x_{ISI}^A . Analogously $x_{\text{ISI}}^{(2)}$ corresponds to $x_{\text{ISI}}^B(t)$ and equivalently for the remaining quantities. Source: [124] ©

Then the dissimilarity profile for the ISI-distance $I(t)$ is defined as the ratio between the difference of the two instantaneous inter-spike intervals and the largest of the two. Namely,

$$I(t) = \frac{|x_{\text{ISI}}^A(t) - x_{\text{ISI}}^B(t)|}{\max\{x_{\text{ISI}}^A(t), x_{\text{ISI}}^B(t)\}},$$

which is a piecewise constant function with discontinuities at the spike times contained in both t^A and t^B . Notice that the quantity $I(t)$ we just introduced ranges from zero (identical ISIs in t^A and t^B) to approximately one (one ISI much larger than the other one). As a consequence, the **ISI-distance** defined by

$$D_I(t^A, t^B) = \frac{1}{T_f} \int_0^{T_f} I(t) dt$$

is a $[0, 1]$ -valued metric which attains the value zero for two spike trains with

the same profile but with a possible time-lag.

As for Kreuz et al.'s SPIKE-distance, it was first proposed in [128] and then refined in [129] to allow a more flexible framework in which a spike time is compared against its closest counterpart in the other spike train. For the latter refined version, new definitions are required. Given $t \in [0, T]$, we already defined the preceding and following spike times in both spike trains, i.e. $t_P^A(t)$, $t_F^A(t)$, $t_P^B(t)$, and $t_F^B(t)$. These four spike times are named the **corners** of time t . For each corner, the distance to the nearest spike in the other train is considered, i.e.

$$\Delta t_P^A(t) = \min_{t_i^B \in t^B} \{|t_P^A(t) - t_i^B|\},$$

and analogously for $t_F^A(t)$, $t_P^B(t)$, and $t_F^B(t)$. Then, for $X = A, B$, each inter-spike interval $x_{\text{ISI}}^X(t)$ is split in the two t -punctuated subintervals of amplitudes

$$x_P^X(t) = t - t_P^X(t), \quad x_F^X(t) = t - t_F^X(t),$$

respectively, so that $x_{\text{ISI}}^X(t) = x_F^X(t) + x_P^X(t)$. Such subinterval amplitudes are then used to compute the following weighted sum for spike train t^A (for the sake of readability we drop the time dependence in the quantities appearing in the r.h.s.)

$$\begin{aligned} S^A(t) &= \left(\frac{\Delta t_P^A}{x_P^A} + \frac{\Delta t_F^A}{x_F^A} \right) / \left(\frac{1}{x_P^A} + \frac{1}{x_F^A} \right) \\ &= \frac{\Delta t_P^A x_F^A + \Delta t_F^A x_P^A}{x_{\text{ISI}}^A}. \end{aligned}$$

The t^B -counterpart, $S^B(t)$, is defined analogously. Finally, the dissimilarity profile for the SPIKE-distance $S(t)$ is defined as a weighted and normaloptimised sum of the contribution from the two spike trains

$$S(t) = \frac{S^A(t)x_{\text{ISI}}^A(t) + S^B(t)x_{\text{ISI}}^B(t)}{2\langle x_{\text{ISI}}(t) \rangle^2},$$

where $\langle x_{\text{ISI}}(t) \rangle = (x_{\text{ISI}}^A(t) + x_{\text{ISI}}^B(t))/2$ is the mean inter-spike interval. Note that such a dissimilarity profile is a piecewise linear function bounded in $[0, 1]$. Analogously to the ISI-distance, the **SPIKE-distance** is given by the temporal average of the dissimilarity profile

$$D_S(t^A, t^B) = \frac{1}{T_f} \int_{t=0}^{T_f} S(t) dt, \quad (7.2)$$

and it is consequently bounded in $[0, 1]$ as well. Note that the complex adaptation introduced in the definition of D^{SPIKE} makes it particularly appealing for

practitioners who deal with spike trains which exhibit multiple time patterns (a fast activity within an overall slower rhythm, for instance) or whose main time scale is not known in advance. In this latter case, the computational load is considerably lightened by avoiding the assessment of the time-scale parameter beforehand.

As for the ambiguity regarding the very first and last spikes, in the case of SPIKE-distance inserting two fictitious action potentials at $t = 0$ and $t = T_f$ introduces an undesired spurious synchronization. For such reason, the authors later proposed a different strategy to deal with such edge issue. We refer to [125] for details in this respect and for further information about variants of the SPIKE-distance.

To conclude this section, we present the SPIKE synchronization, the last metric proposed by Kreuz et al. in [125].

7.3.2 SPIKE synchronization

Such measure shadows the event synchronization coincidence measure described in [172] and, as such, it requires the introduction of a coincidence time window. Such time window is a local maximal distance τ_{ij}^{AB} such that, if two spikes $t_i^A \in t^A$ and $t_j^B \in t^B$ occur in a time interval smaller than τ_{ij}^{AB} (i.e. if $|t_i^A - t_j^B| < \tau_{ij}^{AB}$), then the two spikes are considered coincident. In the SPIKE synchronization, the coincidence interval is adaptively set to be

$$\tau_{ij}^{AB} = \frac{1}{2} \min\{t_{i+1}^A - t_i^A, t_i^A - t_{i-1}^A, t_{j+1}^B - t_j^B, t_j^B - t_{j-1}^B\},$$

so that no parameter is introduced and the time scale is automatically adapted to local firing rate of both spike trains. Then, the coincidence indicator C_i^A defined by

$$C_i^A = \begin{cases} 1 & \text{if } \exists j : |t_i^A - t_j^B| < \tau_{ij}^{AB} \\ 0 & \text{otherwise} \end{cases},$$

flags whether the spike t_i^A is in the coincidence interval of some spike $t_j^B \in t^B$. The coincidence indicator C_j^B can be defined analogously. Using these two indicators, the **SPIKE synchronization** can be defined as

$$S_C(t^A, t^B) = \left(\sum_{i=1}^{\#A} C_i^A + \sum_{j=1}^{\#B} C_j^B \right) / \left(\#A + \#B \right).$$

It is easy to verify that such quantity sums to one if all spikes have a coincident counterpart in the other train, whereas it is zero when no spike has. As a consequence, just as Haas-White and the Schreiber measures, the SPIKE

Name	Notation	Bounds
Cost-based metrics		
Victor-Purpura SPIKE distance	D_q^{SPIKE}	$[\#A - \#B , \#A + \#B]$
Victor-Purpura ISI distance	D_q^{ISI}	\mathbb{R}^+
Embedding-based metrics		
van Rossum distance	D_q^{Ross}	\mathbb{R}^+
Parameter-free metrics		
ISI-distance	D_I	$[0, 1]$
SPIKE-distance	D_S	$[0, 1]$
SPIKE synchronization	D_{SYNC}	$[0, 1]$

Table 7.4: Summary of the main spike-train metrics described in this section, along with the bounds of each distance. Legend: ISI=inter spike interval, $\mathbb{R}^+ = [0, +\infty)$, and $\#A$ and $\#B$ denote the number of action potentials in spike trains A and B , respectively.

synchronization $S_C(\cdot, \cdot)$ is a similarity measure which can be turned into a proper $[0, 1]$ -valued distance by considering its reciprocal

$$D_{\text{SYNC}}(t^A, t^B) = 1 - S_C(t^A, t^B).$$

Note that in Part III, whenever parameter-free spike metrics are employed, we take advantage of the implementation included in the `pyspike` Python module (version 0.5.1) [160], available at [7]. On the other hand, cost-based and embedding-based metrics are computed employing the `elephant` (acronym for electrophysiology analysis toolkit) Python module (version 0.3.0) available at [8].

7.4 Population extensions

We conclude the chapter by highlighting that some of the above mentioned metrics can be generalised to populations of neurons. In particular in [18] a generalization of Victor-Purpura distance is proposed, whereas [93] gives a generalization of the van Rossum distance and [208] a generalization of the embedding-based metrics. As for time-scale adaptive metrics, the neural-ensemble ISI-distance is defined in [127], whereas in [125] a population version for both the SPIKE-distance and the SPIKE synchronization is provided.

Part III

Results on data assimilation experiments

Introduction to Part III

In the third and concluding part of this thesis, we present the results obtained by applying some of the data assimilation methods described in Part I to the single-neuron models illustrated in Part II. Most of the experiments we performed are synthetic, meaning that we tested the methods with the sure knowledge that the prior model is able to reproduce the observations, since it is the one which produced the artificial dataset. However, in the last chapter we also try to assimilate experimental data into a realistic neural model.

Let us start off with the general introduction about the philosophy behind the design of each experiment. We conceived the twin experiments sequentially, moving to next one only when the preceding one was implemented and satisfactorily concluded. The idea was to start with a simple estimation job in a twin-experiment setting to familiarize with some of the studied DA methods. In particular, we wanted to assess and compare some filters effectiveness in tracking down the hidden state variables dynamics, but more importantly to estimate the model parameters and predicting the “future” behaviour of the model. In addition, we were also interested in evaluating the algorithms’ efficiency in order to assess the applicability in large-dimensional models. Keeping in mind our final goal (assimilating real experimental data in a given realistic single-neuron model), we then moved to increasingly challenging assimilation tasks by introducing some obstacles to parameter estimation one after the other. Morally, we wanted to perform parameter assessment with the aim of predicting the out-of-sample model behaviour in more and more realistic conditions, while remaining in a controlled environment. Only after these progressively difficult preliminary tests, we moved to assimilating experimental data into a realistic neuron model, taking advantage of the insight each twin experiment gave us.

In practice, we started from the single-neuron toy model presented in Section 5.2, and on such model we tested three filtering algorithms: two sequential Monte Carlo methods (the bootstrap filter and the optimal sequential importance resampling) and the ensemble Kalman filter. The experiment settings and its results are presented in Chapter 8. In particular, the experiment ob-

jectives were: *i*) estimating the hidden variables dynamics when presenting the same stimulus which generated the data, as well as the model parameters, with high accuracy (see Section 8.2); *ii*) using the estimated parameter values to predict the neuron response to new stimuli the DA algorithm is not aware of (Section 8.3). We report here that we found the ensemble Kalman filter to be the most accurate among the filters in all tasks we were interested in. Specifically, we found it is able to yield to a parameter estimation accuracy which is almost as good as the state-of-the-art variational method minAone, while requiring a significantly smaller computational load.

Then, in the following twin experiments we describe in Chapter 9, we took a more detailed model into account, namely the morphological neuron model described in Section 5.3. This has been considered in the Blue Brain Project to simulate all pyramidal cells of cortical layer two and three (see the supplementary material of [149] and the NMC portal [173]). One of the main motivations was that, thanks to the collaborative philosophy of the Blue Brain Project and direct contacts with the Neuroinformatics division, we had wide access to its sophisticated models and, partially, to its experimental datasets. However, many technical issues arose when using the BBP model. In fact, Neuron is a relatively closed simulation environment which is conceived to be easily usable by neurobiologists rather than to be toggled with for specific mathematical analyses. For instance, we incurred very soon in some possible bugs which occur when some of the activation variables became negative or larger than one. Due to the relatively complex structure of Neuron, which integrates different old-fashioned programming languages, it was difficult to locate the origins of such issues, and we had to enforce the physical bounds on state variables in order to implement a Gaussian state-space model. In particular, we adopted the transformation approach described in Section 4.3, which allows to also impose the positivity of the maximal conductances.

Apart from considering real-valued transformed variables and parameters rather than physical ones, in the first experiment (Section 9.1) we essentially tested the same settings as in the toy-model twin experiment (see Section 9.1.1), but only applied the filter we previously identified to be the most effective in that case. What we wanted was to select and then justify the setting values of the EnKF that in the toy-model case were informally identified as the ones that allowed the three filters to obtain reasonable estimates. In particular, since multi-compartmental models require large computational load to be executed, we wished to reduce the EnKF ensemble size if possible. Then, focusing exclusively on the signal prediction quality, we ran several ANOVA tests (Section 9.1.3) which highlighted that $N = 100$ particles are enough to obtain the best possible outcomes in the conditions we put ourselves in.

Then, we also noted that we always used initial parameter guesses centred on the true parameter values. This means that the resulting parameter estimates are good, but actually worse than the initial guesses. As a consequence, in the second twin experiment (Section 9.2) we considered a biased initial conditions for the parameters, i.e. random initial means for the parameter values 35 times larger than the true ones, in average. Our results showed that it is still possible to obtain good out-of-sample predictions with the EnKF, but there are tolls to pay. The first one is that the mean estimation results are not particularly precise, and only a bunch of good solutions give satisfactory predictions (Section 9.2.1). The second one is that the parameter estimates become less accurate, and that good predictions not necessarily come from accurate parameter estimations (see Section 9.2.2). Nonetheless, we were still confident the EnKF would have allowed us to predict some exact potential profiles with higher accuracy than the parameter values selected by the Blue Brain Project. Indeed, rather than reproducing specific single traces, the BBP estimation procedure aims at fitting the statistics of several electrophysiological features aggregated from different cells. As we discuss in the final Chapter 10, our optimistic perspective was not confirmed by the outcomes of our subsequent investigation.

In fact, we then proceeded with the objective of assimilating the BBP experimental data into the same model used in the twin experiments. Thanks to the direct help and support by S. Jimenez, S. Kerrien, C. Rössert, and W. Van Geit from the project’s Neuroinformatics division, we received and previewed the dataset that was used in the BBP estimation procedure for all pyramidal cells in the microcircuit (i.e., the same step-current recordings as described in Section 10.1)³. After selecting some suitable potential traces, we then proceeded in applying the EnKF using similar settings to those applied in the last twin experiment. As we mention in Section 10.2, these conditions did not allow the EnKF to obtain any estimate of the assimilated true signal, not to mention predicting out-of-sample behaviours. In addition, subsequent consecutive attempts to make the EnKF conditions more favourable were vain. In particular, we first made the parameter search space bounded, and then also further reduced its size (see Section 10.2.2), but this was not enough to obtain good estimates from the EnKF. However, this work was not useless. Indeed, we employed such reduction in the last partly-successful data assimilation we performed.

In fact, since the filtering method we tested appeared not to be usable, we decided to resort to a brute-force minimization method. In particular, thanks

³At the present date, it appears these data are still not publicly available on the NMC portal.

to an efficient Python library developed by researchers at the European Space Agency, we were able to exploit the notion of spike trains distance in order to fit both in-sample and out-of-sample experimental traces (see Section 10.4). In the concluding Section 10.5, we propose some possible explanation of why the EnKF failed and analyse the consequences on the applicability of Bayesian DA methods on neuronal models in relation to the available experimental data.

Chapter 8

Twin experiment on the single-neuron toy model

In this chapter we describe the results of a twin experiment we carried out on the Morris-Lecar-like single-neuron single-compartment model (5.8) introduced in Section 5.2. Recall that in twin experiments, instead of using experimental recording, noisy data are generated from the same mathematical model which is then used to perform data assimilation. This guarantees a controlled environment where only DA method performance is tested rather than specific model-dynamics suitability for a given dataset.

For the sake of readability, we report the model formulation as an ODEs system, namely

$$\begin{cases} \dot{V} &= \left[-\bar{g}_K a(V - E_K) - \bar{g}_{Na} b_\infty(V)(V - E_{Na}) \right. \\ &\quad \left. - \bar{g}_L(V - E_L) + I_{\text{ext}}(t) \right] / C, \\ \dot{a} &= \frac{a_\infty(V) - a}{\tau_a}, \end{cases} \quad \forall t \in [0, T_f].$$

Since we are interested in estimating not only the state variables $V(t)$ and $a(t)$ but also some relevant modelling parameters, for such model we adopt the augmented state-space model (4.7)-(4.8) discussed in Example 4.3. In particular, the dataset for the twin experiment is the quantity $y_{1:J} = \{y_j\}_{j=1}^J$ defined in Section 5.2.1 (recall that for this model we assume no measurement is taken at $j = 0$), and whose first 200 ms are pictured in Figure 5.5. In addition, the true solution underlying the data $y_{1:J}$ is $x_{0:J}^\dagger = (V_{0:J}^\dagger, a_{0:J}^\dagger)^T$, where the length of the discretized data assimilation time window is $J = 50\,000$. This implies that the true set of parameters θ^\dagger is the one given in Table 5.4, where $\theta = (\bar{g}_{Na}, E_{Na}, \bar{g}_K, E_K, \bar{g}_L, E_L, K^{(a)}, V_{1/2}^{(a)}, K^{(b)}, V_{1/2}^{(b)})^T$, whereas parameters C and τ_a are assumed to be fixed and known.

This chapter is structured in the following way. First, in Section 8.1 we describe the twin experiment setting, consisting in the list of applied assimilation

algorithms, the values of their parameters, and the software and the hardware used to implement and then run the methods. Then, Section 8.2 illustrates the filters estimation output both graphically and quantitatively. The ability of the resulting parameter estimates to reproduce an out-of-sample trajectory is tested in Section 8.3, while the final Section 8.4 presents the conclusion of the first twin experiment we performed.

8.1 Twin experiment design

Since we want to test and compare the results of different data assimilation methods, we consider the following Bayesian filtering methods:

- i)* the ensemble Kalman filter (EnKF) summarised in Algorithm 3.4,
- ii)* the bootstrap particle filter (BF) presented in Algorithm 3.5,
- iii)* and the variant optimal importance resampling (OPT-SIRS) given in Algorithm 3.6.

For these three filters we employ an ensemble size $N = 2000$ and unbiased initial condition mean, i.e. $\mu_{x_0} = (V_0^\dagger, a_0^\dagger)^T$ and $\mu_{\theta_0} = \theta^\dagger$. Moreover, the initial covariance matrices is set to be $C_{x_0} = \text{diag}(25 \text{ mV}^2, 0.1)$ and $C_{\theta_0} = 25 I_{d_\theta}$ (units of measure as in Table 5.4), where in general I_d stands for the $d \times d$ identity matrix. Random dynamical noises of the BF and the OPT-SIRS is designed to have the same covariance matrix $\Sigma_x = 10^{-4} I_{d_x}$, and $\Sigma_\theta = 10^{-5} I_{d_\theta}$. On the other hand, for the EnKF $\Sigma_x = 10^{-6} I_{d_x}$, and $\Sigma_\theta = 10^{-6} I_{d_\theta}$.

In what follows, we address the analysis of the results these data assimilation methods yielded. Note that the ensemble Kalman filter, the bootstrap filter and the optimal sequential importance resampling were implemented in MATLAB[®] (version 2015b, The MathWorks Inc., Natick, Massachusetts). In particular, the results described in the following sections are obtained by running 100 instances of every filtering algorithm. Clearly, in each run all random variables involved in the method are sampled independently. The hardware used is a Dell[®] PowerEdge T630 Tower Server with two ten-core Intel[®] Xeon[®] E52650 v3 CPUs and 128GB RAM running Ubuntu Server 14.04.

In addition, we also compare the parameter estimates of the three filters to a state-of-the-art variational method. Specifically, 25 independent runs of minAone (which is implemented in a series of Python scripts invoking the IPOPT optimiser, as described in Section 2.2.2) are launched with the constraint that the parameter vector has to lie inside the hypercube centered at θ^\dagger and having edge length 10 (independently of the unit of measure of the component).

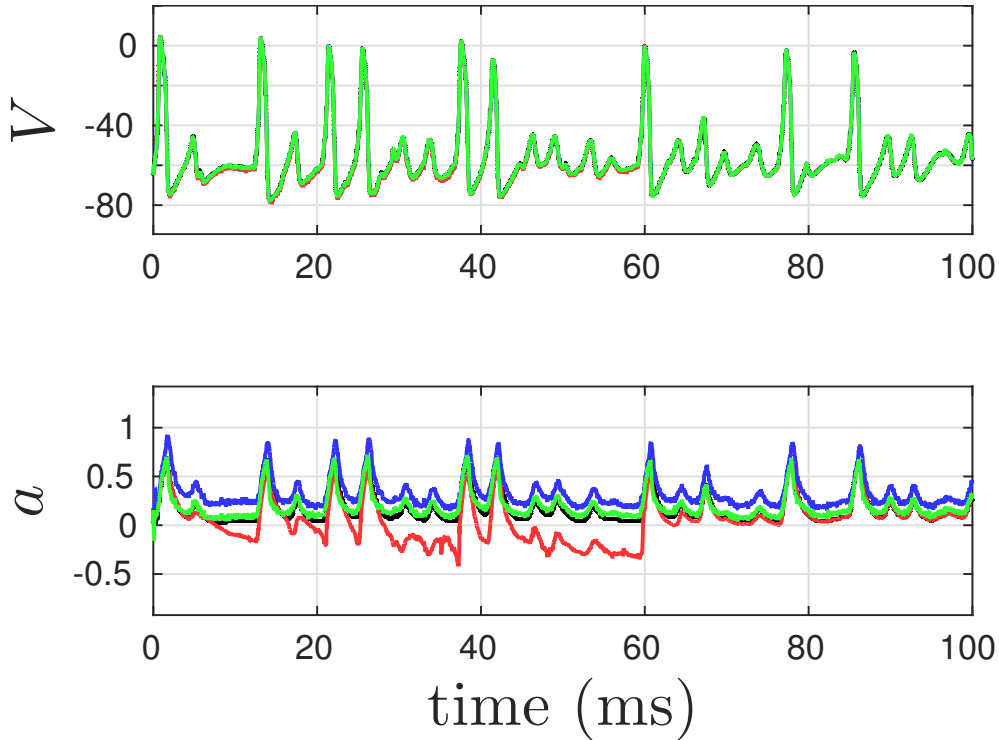


Figure 8.1: True trajectory x_j^\dagger (black line) and variables component of filtering mean μ_{x_j} for the ensemble Kalman filter (red line), the bootstrap filter (blue line) and the optimal sequential importance resampling (green line). Membrane potential V is measured in millivolt (mV).

8.2 Signal and parameters estimation

Figure 8.1 illustrates each filter’s performance by plotting the variables component of the filtering mean in a representative run. Note that the plot is restricted to the first 100 ms of $[0, T_f]$.

As shown in the top panel, in a typical run the true membrane potential V^\dagger substantially overlaps the mean values of all DA methods, which implies that the three filters can recover the true membrane potential values with good accuracy. However, in the lower panel of Figure 8.1, the true unobserved state variable a^\dagger is well estimated by the OPT-SIRS (green line) from the very beginning of the time window $[0, T_f]$, but precisely estimated by the EnKF only after the first 60 ms (red line), and by the BF only after 200 ms in the time window (not visible here). This suggests that the time window length J plays an important role, affecting the estimation quality of unobserved variables.

An equivalent plot representing the filtering mean of the parameter components in the data assimilation time window is given in Figure 8.2. However, in

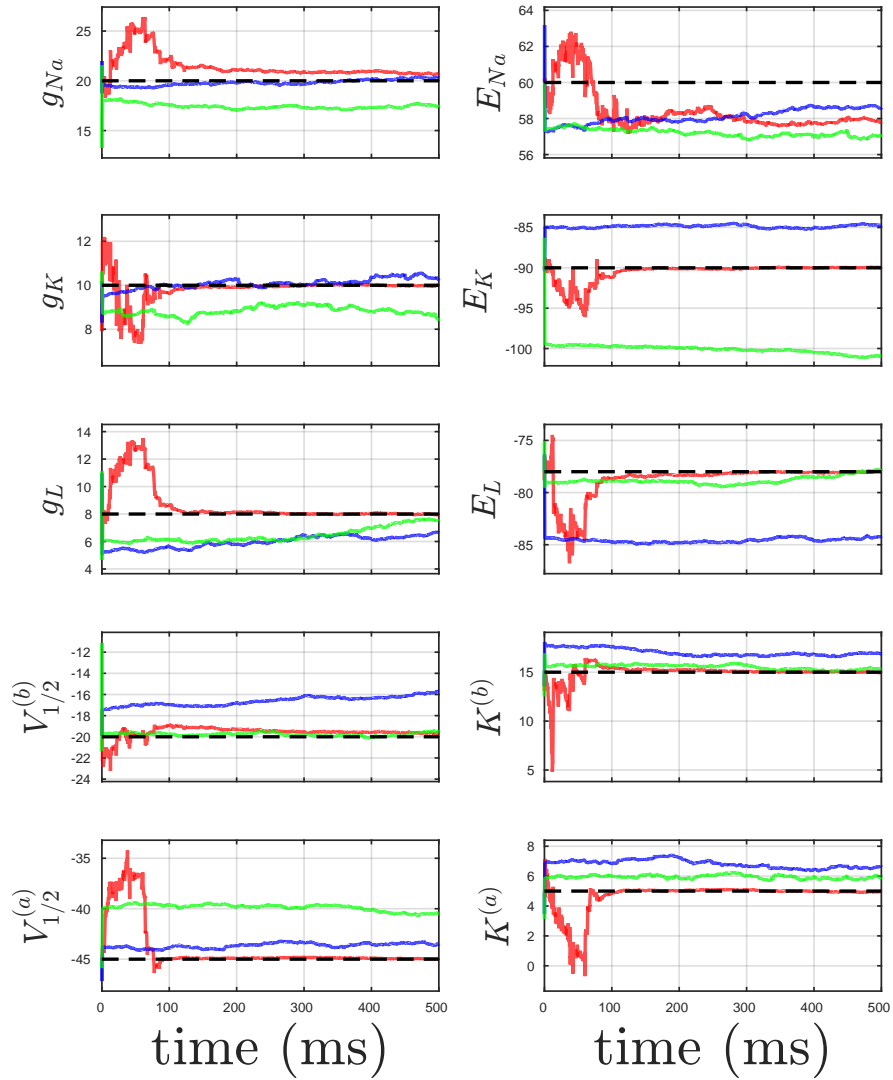


Figure 8.2: True parameter values θ^\dagger (black dashed lines) and parameter component of the filtering mean $\mu_{\theta_{0:j}}$ for the ensemble Kalman filter (red line), the bootstrap filter (blue line) and the optimal sequential importance resampling (green line).

our perspective it is more informative to investigate the parameter component of the corresponding filtering error $|\mu_{\theta_j} - \theta^\dagger|$. Such quantity is represented in Figure 8.3 across the whole data assimilation window $[0, T_f]$. The picture shows

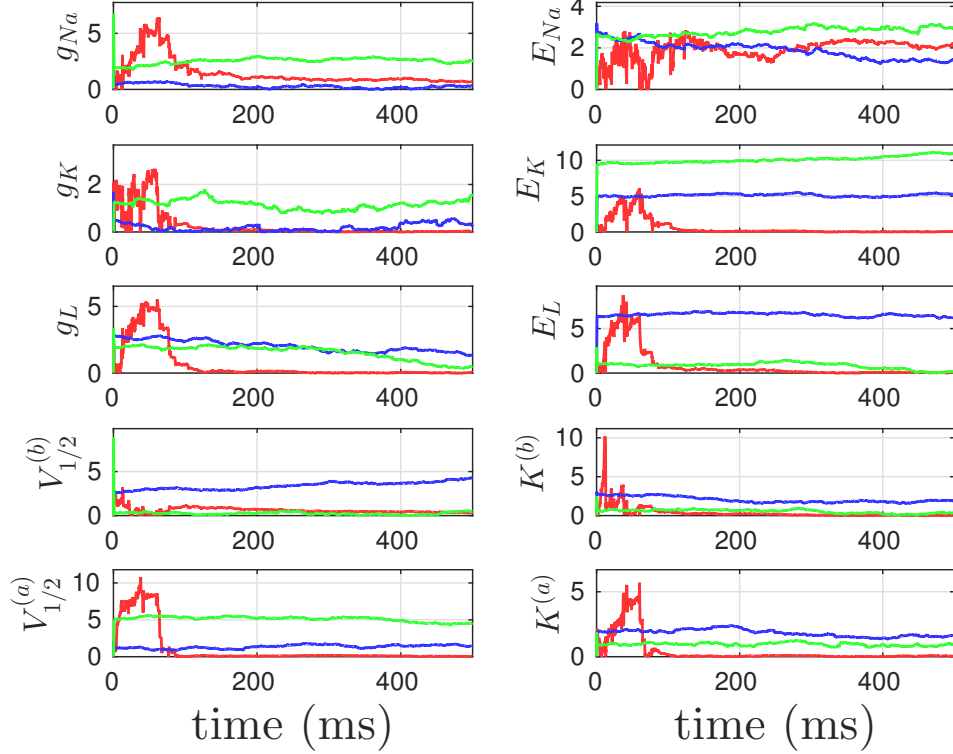


Figure 8.3: Parameter component of the filtering error $|\mu_{\theta_j} - \theta^\dagger|$ for the ensemble Kalman filter (red line), the bootstrap filter (blue line), and the optimal sequential importance resampling (green line).

that the errors of almost all parameters estimated by the EnKF are relatively large at the beginning (essentially in the first 100 ms), but they do approach zero by the end of the data assimilation window. For neural parameters \bar{g}_L , E_L , $V_{1/2}^{(b)}$, $K^{(b)}$, and $K^{(a)}$, the OPT-SIRS produces parameter errors comparable to the EnKF, but the errors for the other parameters are larger. However, in this run the BF gives the lowest performance with only few parameter errors tending to zero.

In addition, it can be remarked that filtering parameter errors usually pass through some initial transient states, and then eventually stabilize on some asymptotic value. This fact is exploited in order to get a scalar estimate of every parameter by an average estimator

$$\hat{\theta} = \frac{1}{J - J_1 + 1} \sum_{j=J_1}^J \mu_{\theta_j}. \quad (8.1)$$

We take $J_1 = 35\,000$ iterations in practice, i.e. the average is performed over the last three tenths of $[0, T_f]$, which allows one to discard the initial transient

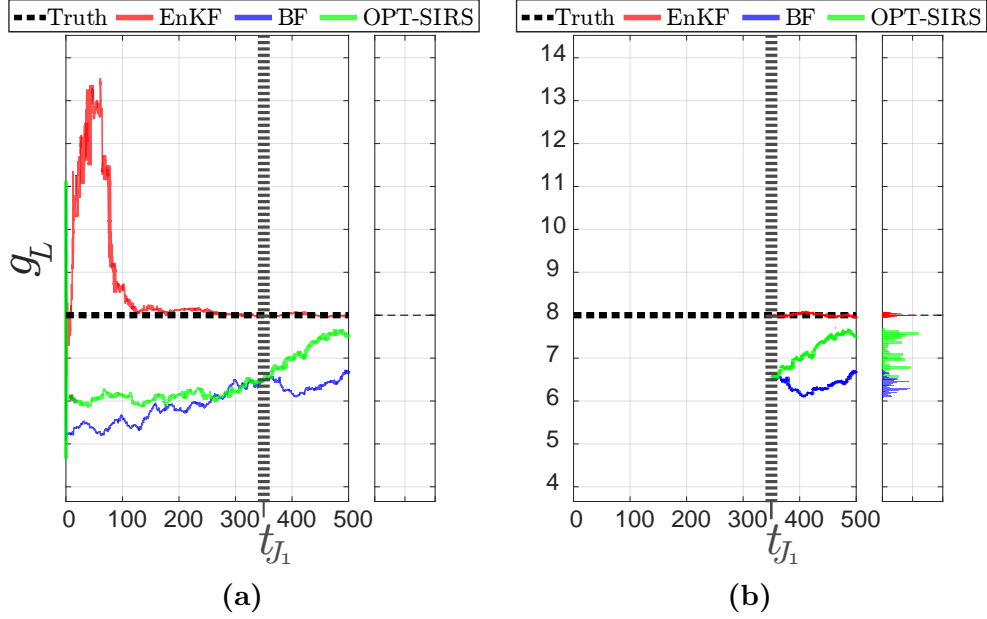


Figure 8.4: Graphical representation of (8.1) (the definition of $\hat{\theta}$) for component \bar{g}_L . (a) posterior distribution mean $\mu_{\theta_{0:J}}$ for component \bar{g}_L (detail from Figure 8.2); and (b) posterior distribution mean μ_{θ_j} for $j \in \{J_1, \dots, J\}$ with corresponding empirical histograms $\frac{1}{J-J_1+1} \sum_{j=J_1}^J \delta_{\mu_{\theta_j}}$ (right panel).

states. Such averaging procedure is illustrated in Figure 8.4a, which represents the filtering mean for parameter \bar{g}_L along with a vertical dashed grey line placed at time $t_{J_1} = J_1 \Delta t$. Then, the empirical distribution $\frac{1}{J-J_1+1} \sum_{j=J_1}^J \delta_{\mu_{\theta_j}}$ resulting from the values of the filtering mean in the interval $\{J_1, \dots, J\}$ is plotted in the right panel of Figure 8.4b. We highlight that (8.1) is exactly the mean of such empirical distribution.

A more comprehensive analysis of the estimation results is available in Table 8.5, where the estimated parameters means and standard deviations are listed. In the first three columns, such values are computed by evaluating the sample mean and standard deviation from the 100 independent runs we launched for every filtering method. In addition, the last column displays the estimated parameter vector provided by minAone.

For every parameter value, we performed a one-sample t-test to check whether the difference between the mean estimate and the corresponding true value is statistically significant. Our results proved that for EnKF this is indeed the case (for all parameters $p < 5\%$). On the other hand, the same tests for both the BF and the OPT-SIRS showed that this difference is statistically significant only for the four (out of ten) parameters \bar{g}_K , \bar{g}_L , $K^{(b)}$, $K^{(a)}$.

$\hat{\theta}$		EnKF	BF	OPT-SIRS	minAone
\bar{g}_{Na}	mean	18.56	20.74	19.86	20.77
	SD	1.41	4.40	3.80	
E_{Na}	mean	61.43	60.23	59.99	58.86
	SD	2.11	4.76	5.00	
\bar{g}_{K}	mean	9.99	12.97	12.41	9.99
	SD	0.06	3.55	3.89	
E_{K}	mean	-89.90	-90.62	-90.17	-90.00
	SD	0.25	4.73	4.72	
\bar{g}_{L}	mean	7.65	7.14	6.50	8.10
	SD	0.35	3.72	3.10	
E_{L}	mean	-77.27	-78.58	-77.31	-78.25
	SD	0.63	4.41	4.83	
$V_{1/2}^{(b)}$	mean	-20.77	-20.89	-19.96	-19.58
	SD	0.75	5.02	4.83	
$K^{(b)}$	mean	14.61	16.84	16.24	15.14
	SD	0.35	2.53	2.72	
$V_{1/2}^{(a)}$	mean	-45.01	-44.65	-44.34	-45.00
	SD	0.05	5.02	4.87	
$K^{(a)}$	mean	4.92	5.92	6.77	5.00
	SD	0.05	4.20	4.07	

Table 8.5: Mean and standard deviation of estimated parameters, with respect to the 100 independent runs of each filter. The last column shows the parameter estimates for the best solution provided by minAone in its 25 independent runs (i.e., the one with minimal cost-function value)

As for the variability, the standard deviations of EnKF estimates are significantly smaller than those of BF or OPT-SIRS. This is further confirmed by observing that the average coefficient of variation¹ of the EnKF ($\overline{CV}_{\text{EnKF}} = 0.024$) is one order of magnitude smaller than both the BF ($\overline{CV}_{\text{BF}} = 0.240$) and the OPT-SIRS coefficient of variation ($\overline{CV}_{\text{OPT}} = 0.230$). In fact, the large difference in standard deviations could explain the result of the t-tests.

Table 8.6 further investigates the matter of parameter estimation accuracy by presenting, for every parameter, the average relative errors for each method. Overall, the EnKF performs substantially better than both particle filters. In fact, the average mean relative error of the EnKF is about 8 times smaller than both BF's and OPT-SIRS's average error. Using a non-parametric Fried-

¹ The coefficient of variation (CV) of a random variable with mean μ and standard deviation σ is defined as $CV = \frac{\sigma}{|\mu|}$

$ \hat{\theta} - \theta^\dagger / \theta^\dagger $	EnKF	BF	OPT-SIRS	minAone
\bar{g}_{Na}	$8.28 \cdot 10^{-2}$	$1.79 \cdot 10^{-1}$	$1.43 \cdot 10^{-1}$	$3.83 \cdot 10^{-2}$
E_{Na}	$3.34 \cdot 10^{-2}$	$6.10 \cdot 10^{-2}$	$6.98 \cdot 10^{-2}$	$1.90 \cdot 10^{-2}$
\bar{g}_{K}	$3.90 \cdot 10^{-3}$	$3.62 \cdot 10^{-1}$	$3.65 \cdot 10^{-1}$	$8.51 \cdot 10^{-4}$
E_{K}	$1.99 \cdot 10^{-3}$	$4.38 \cdot 10^{-2}$	$4.29 \cdot 10^{-2}$	$6.56 \cdot 10^{-6}$
\bar{g}_{L}	$5.12 \cdot 10^{-2}$	$3.82 \cdot 10^{-1}$	$3.47 \cdot 10^{-1}$	$1.30 \cdot 10^{-2}$
E_{L}	$1.04 \cdot 10^{-2}$	$4.46 \cdot 10^{-2}$	$5.19 \cdot 10^{-2}$	$3.26 \cdot 10^{-3}$
$V_{1/2}^{(b)}$	$4.36 \cdot 10^{-2}$	$2.05 \cdot 10^{-1}$	$1.89 \cdot 10^{-1}$	$2.11 \cdot 10^{-2}$
$K^{(b)}$	$2.94 \cdot 10^{-2}$	$1.72 \cdot 10^{-1}$	$1.54 \cdot 10^{-1}$	$9.39 \cdot 10^{-3}$
$V_{1/2}^{(a)}$	$8.81 \cdot 10^{-4}$	$8.92 \cdot 10^{-2}$	$9.09 \cdot 10^{-2}$	$1.07 \cdot 10^{-4}$
$K^{(a)}$	$1.71 \cdot 10^{-2}$	$6.70 \cdot 10^{-1}$	$6.94 \cdot 10^{-1}$	$4.41 \cdot 10^{-4}$
Average	$2.75 \cdot 10^{-2}$	$2.21 \cdot 10^{-1}$	$2.15 \cdot 10^{-1}$	$1.06 \cdot 10^{-2}$

Table 8.6: Mean of parameter estimation relative error, with respect to the 100 independent runs of each filter. The last column shows the relative error for the best solution provided by minAone in its 25 independent runs (i.e., the one with minimal cost-function value).

man test with the method as column effect and the parameter as row effect, it was proved that the difference between the EnKF's relative error and both particle filters is indeed statistically significant ($p < 5\%$), whereas the difference between the BF and the OPT-SIRS is not. In addition, the EnKF mean errors are at most one order of magnitude larger than those resulting from the minAone run which produced the minimal cost-function error, the only exceptions being E_{K} and $K^{(a)}$.

8.3 Signal prediction results

Now that the parameter estimator $\hat{\theta}$ has been defined and its distribution explored, we want to validate the estimates we obtained by checking whether they can provide good predictions of the model dynamics beyond the original time window $[0, T_f]$.

In practice, we first partition the whole data assimilation time window into two equal intervals, set $T_i := \frac{T_f}{2} = J_i \Delta t$, and then name the resulting second half interval $[T_i, T_f]$ the **generalization time window**. Equivalently, we can call $[T_i, T_f]$ the **in-sample time window**, referring to the fact that we are presenting (part of) the same stimulus which was presented in the data assimilation process. System (5.8) is then solved numerically in such interval using the estimated parameter values $\hat{\theta}$ as modelling parameters and with

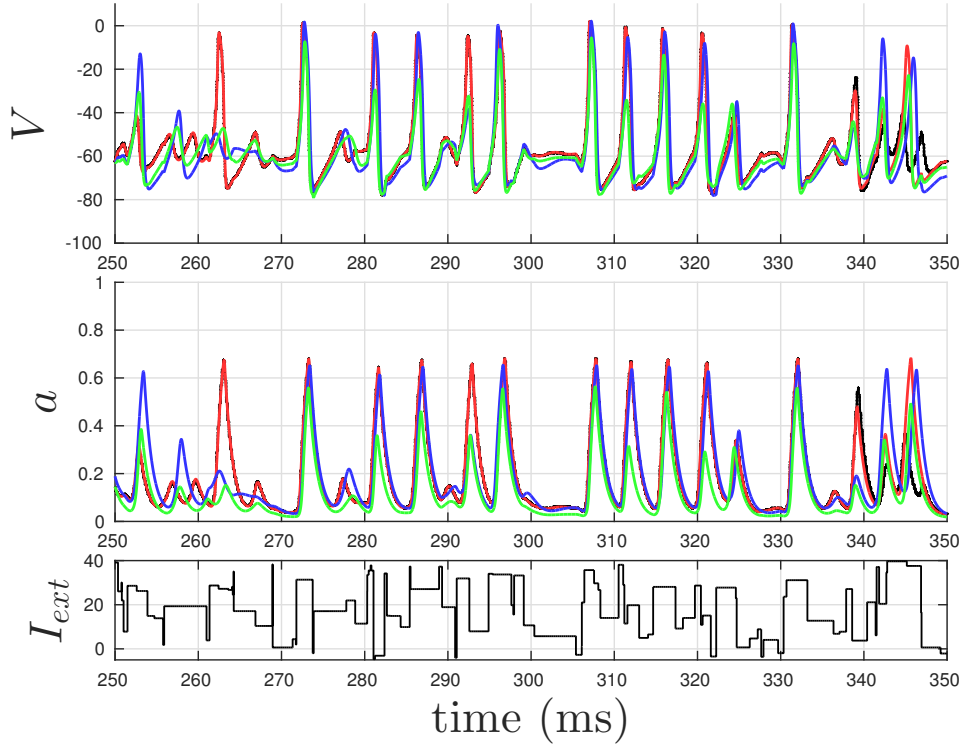


Figure 8.7: Forecast skill in the first 100 ms of the generalization time window $[T_i, T_f]$ of the ensemble Kalman filter (red line), the bootstrap filter (blue line), and the optimal sequential importance resampling (green line) along with the true trajectory (black line in upper panels) and the time-dependent input current $I_{ext}(t)$ (black line in lower panel).

initial data $(V(T_i), a(T_i))^T = \mu_{x_{J_i}}$. We write $\hat{x}_j^{(in)}$ for $j \in \{J_i, \dots, J\}$ to denote the resulting estimated trajectory.

Figure 8.7 shows that the EnKF estimate overlaps the true trajectory almost perfectly ($\hat{x}_{J_i:J}^{(in)} \approx x_{J_i:J}^\dagger$), whereas both the BF and the OPT-SIRS have rather similar profiles which are close to the true trajectory, but not as close as the EnKF.

After verifying that the parameter estimator $\hat{\theta}$ produces a good estimate of the system dynamics within the data assimilation window, its prediction capability was further investigated. Estimates over the generalization time window were continued “in the future” over the **out-of-sample time window** $[T_f, 3T_f]$ (also called **prediction time window**), producing the out-of-sample estimate $\hat{x}_{J:3J}^{(out1)}$.² Then, such estimate was tested against the continuation of the true solution $x_{J:3J}^\dagger$.

²Notice that the prediction time window is twice as long as $[0, T_f]$.

$\sum_j \hat{x}_j - x_j^\dagger \Delta t$		EnKF	BF	OPT-SIRS
in-sample window	V (mean)	223.3	4783.4	4576.1
	a (mean)	2.2	59.2	54.6
out-of-sample window	V (mean)	807.3	19075.2	18133.0
	a (mean)	7.8	236.8	217.7

$\sum_j \hat{V}_j - V_j^\dagger / \sum_j (\hat{V}_j - V_j^\dagger + V_j^\dagger - y_j)$		EnKF	BF	OPT-SIRS
in-sample window	mean	52.21%	95.38%	95.13%
	st. dev.	8.62%	1.69%	1.80%
out-of-sample window	mean	48.97%	95.35%	95.10%
	st. dev.	8.33%	1.73%	1.82%

Table 8.8: Mean L^1 -error in generalization and prediction time windows (top panel) and relative estimation error (lower panel)

Quantitative measures of how close estimated trajectories are to the true one are presented in Table 8.8. These include the mean L^1 -error³ for each variable in the in-sample window (upper panel, first row), in the out-of-sample window (second row), and the mean and standard deviation of the $d_1^{(\text{rel})}$ -error defined in (4.12) in both in-sample and out-of-sample time windows (lower panel). Recall that the $d_1^{(\text{rel})}$ -distance is a $[0, 1)$ -valued function which tends to one if $d_1(\hat{V}, V^\dagger) \gg d_1(V^\dagger, y)$, and approaches zero if the estimation error is much smaller than the model-intrinsic measurement error. Note that in order to compute the $d_1^{(\text{rel})}$ -error in the out-of-sample time window, a new dummy dataset $y_{J:3J}^{(\text{out1})}$ is generated from the continuation of the true solution $x_{J:3J}^\dagger$ by adding a Gaussian noise $\mathcal{N}(0, \Gamma)$.

The top panel in Table 8.8 shows that the EnKF presents a cumulative mean $L^1([T_i, 3T_f])$ -error of variable V which is $1/22$ of the OPT-SIRS and $1/23$ of the BF. In addition, the OPT-SIRS presents a mean L^1 -error of variable a which is smaller than the BF in both generalization and prediction time window, but still one order of magnitude larger than the EnKF. Nonetheless, no statistically significant difference between the BF and the OPT-SIRS was detected applying the Friedman test.

³ The $L^1([T_i, T_f])$ -error of the membrane potential component over the in-sample time window is defined as $d_1(\tilde{V}, V^\dagger) = \int_{T_i}^{T_f} |V(t) - V^\dagger(t)| dt \approx \sum_{j=J_i}^J |\tilde{V}_j - V_j^\dagger| \Delta t$. The last discrete sum shows the way this error is actually computed and it represents the value of the integral approximated by the Euler integration method. Note that it is equal to the d_1 -distance (4.11) scaled by Δt . $L^1([T_i, T_f])$ -errors for variable a and $L^1([T_f, 3T_f])$ -errors in the out-of-sample time window are defined analogously.

Remarkably, in the lower panel, the first column shows that the EnKF mean relative error is approximately 0.5. This means that the L^1 -error is as large as the truth-data error in average, so that the EnKF produces such a good estimate that the mean L^1 -error is comparable to the measurement error. In addition, not only this is true in the in-sample window, but also in the out-of-sample time window, even though the dummy dataset in the out-of-sample time window was not used in the DA methods application. The relatively small standard deviations prove the robustness of this result. Again, the BF and the OPT-SIRS are hardly distinguishable, with a much larger relative error than the EnKF and a small standard deviation.

8.4 Discussion and conclusions

In this chapter, we compared three filtering data assimilation methods in a problem of simultaneous parameters and unmeasured variables estimation of a neuronal model. This study was performed in a twin experiment setting in which data were artificially generated by numerical simulation of the neural model. Our results demonstrate that the ensemble Kalman filter, the bootstrap filter and the optimal sequential importance resampling are all suitable methods for parameter estimation and they all possess the capability to predict the future activity of a single neuron.

As we saw, *t*-tests showed the mean EnKF estimate is significantly different from the true value. On the other hand, the known fact that particle filters can recover the true filtering distribution in the limit as $N \rightarrow \infty$ is consistent with the BF and the OPT-SIRS being unbiased. Nonetheless, there is no guarantee that a particle filter can provide a better performance in any single realization.

In fact, the EnKF is by far the best of these methods in the more telling task of signal estimation prediction. Both particle filters provide similar results, with the OPT-SIRS performing slightly better than the BF but with no statistically significant difference between them. Besides, further analysis of the parameter estimation performance demonstrated that the EnKF has a much smaller relative error than the two particle filters.

In addition, in Section 8.2 we found hints that the data-assimilation-window length J can be an important factor for estimation accuracy. However, it should be highlighted that in all simulations we ran the performance of the EnKF is robust with respect to J and other preassigned quantities such as μ_{x_0} , μ_{θ_0} , C_{x_0} and C_{θ_0} .

The computation loads were also compared. It was shown that all filtering methods require a similar computing wall time for each run (2 min 47 s \pm 12 s for the EnKF, 2 min 14 s \pm 7 s for the BF, and 2 min 17 s \pm 5 s for the OPT-

SIRS). Note that we took advantage of the automatic parallelization of Matlab 2015b.

It is not surprising that the smoothing method minAone can estimate parameters with very good accuracy, since the whole dataset is used in this variational method. However, the computational time consumption of minAone is much larger than the filtering methods. As we found, a single run took on average 3 h 46 min 20s. Therefore, seeking a trade-off between accuracy of estimates and the computing time efficiency, we conclude that the EnKF is the best choice in this example.

In the following chapters, we further develop the application of Bayesian DA methods in computational neuroscience by investigating the more biologically accurate neuron models presented in Section 5.3.

Chapter 9

Twin experiments on the BBP single-neuron model

In this chapter, we present the results of two numerical experiments we carry out on the multi-compartment neuron model proposed by the Blue Brain Project described in Section 5.3. The aim of such synthetic experiments is to test the ensemble Kalman filter (which in the preceding chapter was found to be the most effective method for parameter estimation in a augmented state-space model framework) on a realistic single-neuron model.

In particular, in Section 9.1 we describe a twin experiment aimed at identifying a good set of the filter parameters. Indeed, selecting suitable setting parameters of the EnKF is of paramount importance to make the method effective while preserving computational efficiency. Consider, in fact, that simulating a neuron behaviour for a few seconds through the complex BBP model takes a non-negligible execution time, so that reducing the ensemble size to its minimal effective size may lead to a considerable speed-up. On the other hand, in the subsequent Section 9.2 we make twin-experiment conditions less favourable by introducing a bias in the parameter initial conditions. This is to inspect whether or not the EnKF is able to estimate parameters even when the initial guess is not centred on the right values, which can give hints about its applicability on more practical problems. In fact, in the following chapter, we take an experimental dataset into account (i.e., true parameters are not known in advance) and we try to assimilate such data in the same BBP neuron model considered in these twin experiments.

Entering in the details, in the following two sections we present the results obtained by assimilating the free parameters of model L23_PC_cADpyr229_1 (namely, those marked with a \checkmark sign in Table 5.7) plus the maximal conductance of the leakage current I_L and the hyperpolarization-activated current I_H . In this case, the target model is the model for the continuous accommodating

pyramidal cell of layer 2 and 3 identified by equations (5.11), (5.12), (5.13), and (5.14). Note that, in this and the following chapter we only address the estimation of state variables and modelling parameters of the somatic compartment. As in the previous chapter, the state-space model adopted is augmented (see page 60), Gaussian, and nonlinear.

Let us now specify which state variables and parameters enter the dynamic component of the signal variable x and the parameter (i.e., static) component θ . Focusing on the soma means that parameter θ has entries given by the modelling parameters listed in the somatic column of Table 5.7, i.e. $\bar{g}_{\text{Nat2}}^{(\text{soma})}$, $\bar{g}_{\text{Kv3.1}}^{(\text{soma})}$, $\bar{g}_{\text{SK}}^{(\text{soma})}$, $\bar{g}_{\text{CaHVA}}^{(\text{soma})}$, $\bar{g}_{\text{CaLVA}}^{(\text{soma})}$, $\bar{g}_{\text{L}}^{(\text{soma})}$, $\bar{g}_{\text{H}}^{(\text{soma})}$, $\gamma^{(\text{soma})}$, $\text{decay}^{(\text{soma})}$. All remaining parameters are considered to be fixed to the values listed in the table. In the same fashion, the random vector X_j has entries given by the noisy version of all somatic variables, i.e. $V^{(\text{soma})}$, $m_{\text{Nat2}}^{(\text{soma})}$, $h_{\text{Nat2}}^{(\text{soma})}$, $m_{\text{Kv3.1}}^{(\text{soma})}$, $m_{\text{CaHVA}}^{(\text{soma})}$, $h_{\text{CaHVA}}^{(\text{soma})}$, $m_{\text{CaLVA}}^{(\text{soma})}$, $h_{\text{CaLVA}}^{(\text{soma})}$, $m_{\text{H}}^{(\text{soma})}$, $z_{\text{SK}}^{(\text{soma})}$, $[Ca^{2+}]_{\text{in}}^{(\text{soma})}$. All state variables for other compartments are kept out in this analysis and their dynamics is assumed to be deterministic.

Note that we employ the transformation-based approach described in Section 4.3. This is to deal with the boundedness of the activation variables, of the maximal conductances, as well as the internal calcium concentration and the parameters involved in its dynamics. We first introduced such precaution because we found that the Neuron implementation of the BBP models crashes when the state variables exit their natural bounds. In addition, enforcing bounds on the model parameters allows the algorithm to explore the search space only in the biologically-meaningful regions. Moreover, when the assimilation task becomes difficult (as we encountered in some preparatory tests on the experimental-data case), we expect that restricting the parameters search space could in principle yield better prediction results.

As a consequence, in the experimental setting described in the current section, the signal variable is given by the stochastic counterpart of $x = \left(V^{(\text{soma})}, \text{logit}(m_{\text{Nat2}}^{(\text{soma})}), \text{logit}(h_{\text{Nat2}}^{(\text{soma})}), \text{logit}(m_{\text{Kv3.1}}^{(\text{soma})}), \text{logit}(m_{\text{CaHVA}}^{(\text{soma})}), \text{logit}(h_{\text{CaHVA}}^{(\text{soma})}), \text{logit}(m_{\text{CaLVA}}^{(\text{soma})}), \text{logit}(h_{\text{CaLVA}}^{(\text{soma})}), \text{logit}(m_{\text{H}}^{(\text{soma})}), \text{logit}(z_{\text{SK}}^{(\text{soma})}), \log([Ca^{2+}]_{\text{in}}^{(\text{soma})}) \right)^T$. On the other hand the parameter is given by $\theta = \left(\log(\bar{g}_{\text{Nat2}}^{(\text{soma})}), \log(\bar{g}_{\text{Kv3.1}}^{(\text{soma})}), \log(\bar{g}_{\text{SK}}^{(\text{soma})}), \log(\bar{g}_{\text{CaHVA}}^{(\text{soma})}), \log(\bar{g}_{\text{CaLVA}}^{(\text{soma})}), \log(\bar{g}_{\text{L}}^{(\text{soma})}), \log(\bar{g}_{\text{H}}^{(\text{soma})}), \log(\gamma^{(\text{soma})}), \log(\text{decay}^{(\text{soma})}) \right)^T$. Note that in the remaining part of this section we quit denoting variable with the “(soma)” superscript, for the sake of readability.

Considering that in Section 4.2 we described how the discrete time map $g_{j-1}(x_{j-1}, \theta_{j-1})$ and its stochastic counterpart $f_{j-1}(x_j | x_{j-1}, \theta_{j-1})$ are constructed

from a given vector flow $F(t, x; \theta)$, now that we explained what are the entries of the augmented state variable $z = (x, \theta)$, we can proceed describing the first twin experiment.

9.1 First experiment: fine tuning of EnKF parameters

In this twin experiment, we focus on identifying the most effective value for some setting parameters of the EnKF algorithm, namely the size of the ensemble N , the dynamical noise for the state variable components Σ_x and for the parameter components Σ_θ . Note that, to the best of our knowledge, there is no straightforward way to select a suitable value for the diffusion constants entering matrices Σ_x and Σ_θ in a given application. The choice of a value for Σ_v is part of the selection of the stochastic model, but in many cases there is no quantitative measure of the noise level. On the other hand, the entries of Σ_θ determine the exploration rate in the parameter space: a large value for Σ_θ may lead to large variability in the parameter estimates or even divergence, while a small value implies slow convergence. As a consequence, it is a common practice to consider them as a further parameter to be estimated [55, 206]. Our experiment aims at avoiding such further extension of the parameter space to only focus on the estimation of the model parameters. In order to do this, we launch a series of independent runs of the EnKF algorithm we implemented in the Neuron simulation environment using its Python interface in a full factorial design.

All programs are launched requesting 8 computing nodes on a cluster with one head node and 14 computing nodes. Each node of the cluster consists in a Dell[®] PowerEdge R730 with two sockets, each mounting E5-2660V3 ten-core Intel[®] Xeon[®] E5-2660v3 CPUs. All nodes mount a 128GB RAM, except for the head node which mounts a 64GB RAM.

The experiment description is structured as follows. In Section 9.1.1 we illustrate the design choices for the above-mentioned setting parameters. Then, Section 9.1.2 describes a sample output of the ensemble Kalman filter, so that in Section 9.1.3 we have some reference results in mind when illustrating the application of a series of ANOVA tests aimed at identifying the best list of setting parameters.

9.1.1 Setting parameters for EnKF

First, we considered dynamical noise covariance matrices of the block-diagonal form

$$\Sigma_x = \begin{bmatrix} \sigma_v^2 & 0 & 0 & 0 \\ 0 & \sigma_m^2 I_8 & 0 & 0 \\ 0 & 0 & \sigma_z & 0 \\ 0 & 0 & 0 & \sigma_{[Ca^{2+}]_{in}}^2 \end{bmatrix} = \begin{bmatrix} \sigma_v^2 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & \sigma_m^2 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & \sigma_m^2 & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_m^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_z & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \sigma_{[Ca^{2+}]_{in}}^2 \end{bmatrix},$$

and $\Sigma_\theta = \text{diag}(\sigma_g^2 I_7, \sigma_\gamma^2, \sigma_{\text{decay}}^2)$, which expanded writes

$$\Sigma_\theta = \begin{bmatrix} \sigma_g^2 I_7 & 0 & 0 \\ 0 & \sigma_\gamma^2 & 0 \\ 0 & 0 & \sigma_{\text{decay}}^2 \end{bmatrix} = \begin{bmatrix} \sigma_g^2 & 0 & \dots & 0 & 0 & 0 \\ 0 & \sigma_g^2 & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_g^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_\gamma^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{\text{decay}}^2 \end{bmatrix}.$$

Such covariance matrices allow us to consider a noise amplitude which is the same among all activation and inactivation variables (σ_m) and among all maximal conductances (σ_g), but differs from the noise amplitude for the membrane potential (σ_v), for the internal calcium concentration variable ($\sigma_{[Ca^{2+}]_{in}}$), and for its modelling parameters γ and decay (σ_γ and σ_{decay} , respectively). In what follows we fix $\sigma_m = \sigma_z = \sigma_\gamma = 1e-2$ and $\sigma_{\text{decay}} = \sigma_{[Ca^{2+}]_{in}} = 1e-3$ and then explore different values of the remaining setting parameters (namely σ_v , σ_g) and the ensemble size N .

The remaining EnKF parameters include those involved in the stochastic initial condition Z_0 . In this particular experimental setting, we start off by considering a Gaussian initial condition centred in the true¹ initial conditions, i.e. $\mu_{x_0} = x_0^\dagger$ and $\mu_{\theta_0} = \theta^\dagger$. The variance matrices for the initial conditions are set to be $C_{x_0} = \text{diag}(c_{v_0}^2, c_{m_0}^2 I_8, c_{z_0}^2, c_{[Ca^{2+}]_{in_0}}^2)$ and $C_{\theta_0} = \text{diag}(c_{g_0}^2 I_7, c_{\gamma_0}^2, c_{\text{decay}_0}^2)$, where $c_{v_0} = 10$, $c_{m_0} = c_{z_0} = c_{g,0} = c_{\gamma_0} = 0.7$, $c_{[Ca^{2+}]_{in_0}} = c_{\text{decay}_0} = 0.1$ which are relatively large values.

¹Meaning those which produced the artificial datasets $y_{J:2J}^{(\text{in})}$, $y_{J:2J}^{(\text{out1})}$ and $y_{J:2J}^{(\text{out2})}$ illustrated in Section 5.3.3.

9.1.2 Sample EnKF output

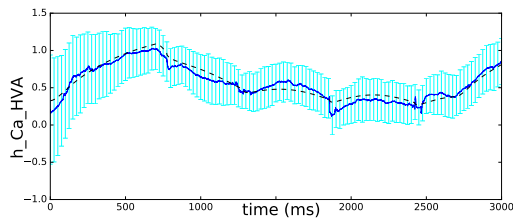
In this section we present the result of a sample run of the EnKF in the setting just discussed, where the ensemble size is $N = 100$ and the remaining dynamical noise parameters are set to be $\sigma_v = 1e-2$, $\sigma_g = 1e-2$.

In Figure 9.1 we picture a graphical representation of the filtering distribution $p(x_{0:j}, \theta_{0:j} | y_{0:j})$, for $j = 0, \dots, J$. Let us first focus on Figure 9.1a, where the dashed black line denotes the h_{CaHVA} -component of the true trajectory $x_{0:J}^\dagger$. The other lines are meant to illustrate the approximated posterior distribution computed by the EnKF: the blue solid line is the (empirical) filtering mean and the cyan solid bands denote the corresponding standard-deviation confidence intervals (i.e. mean \pm standard deviation).

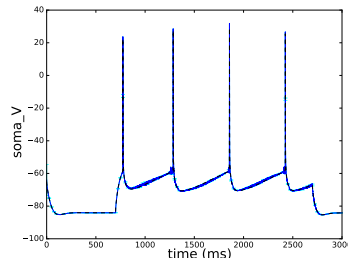
As one can notice, the filtering mean is consistently close to the true trajectory which, in return, always lies in the confidence interval. This suggests that the filtering mean is a reasonable point-wise estimator for the time-evolving true trajectory. In order to verify whether other standard estimators track the true signal down better than the mean, we also inspected the filtering mode (not shown here). However, it was conspicuously visible that the empirical mode is too discontinuous on such a small ensemble, and, what is more, it moves away from the true trajectory more than the mean. This supports the choice of the filtering mean as estimator for the true trajectory over the filtering mode.

Let us move to the other x -components (i.e. state variables). Unsurprisingly, Figure 9.1b shows that the membrane potential is indeed well recovered. In fact, the only measured variable is exactly the noisy voltage of the modelled neuron. The remaining variables plotted in Figure 9.1c are also properly tracked down, as one can deduce from the narrow cyan bands. It looks like the less accurate estimation is the one for the h_{CaHVA} variable in the top-right corner of Figure 9.1c (magnified in 9.1a), but this due to the different range of variable h_{CaHVA} . In fact, while its variation in the logit-scale is of the order $1e-1$, the other variables' ranges are in the unit order.

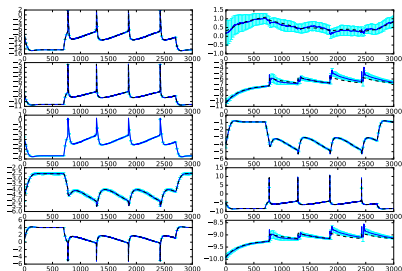
A separate discussion should be done for the static θ -component of the augmented signal variable. In Figure 9.2a we present the filtering distribution for parameter $\bar{g}_{\text{Nat}2}$. Although the true value (dashed black line) always remains in the confidence interval (cyan bands) and the filtering mean (plain blue line) generally hovers around it, we notice some large excursion from the correct value in correspondence of the somatic spikes (compare with Figure 9.1b). This likely to be due to the EnKF-update step (line 11 in Algorithm 3.4). Indeed, a large innovation $\delta_j = y_j - H\hat{z}_j$ caused by a wrongly predicted state in correspondence of an action potential can result in a large correction in the unmeasured variables at the analysis step. In case of parameter $\bar{g}_{\text{Nat}2}$,



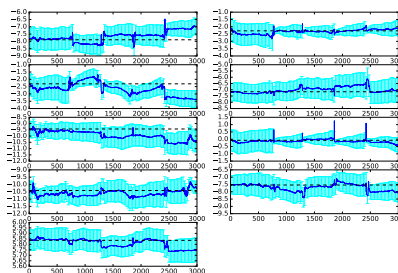
(a) Filtering distribution for variable h_{CaHVA} in a logit scale. Detail from the top right panel in (c).



(b) Somatic membrane potential V .



(c) $\text{logit}(m_{\text{CaHVA}})$, $\text{logit}(h_{\text{CaHVA}})$,
 $\text{logit}(m_{\text{Kv3.1}})$, $\text{logit}(z_{\text{SK}})$,
 $\text{logit}(m_{\text{CaLVA}})$, $\text{logit}(h_{\text{CaLVA}})$,
 $\text{logit}(m_{\text{H}})$, $\text{logit}(m_{\text{Nat2}})$, $\text{logit}(h_{\text{Nat2}})$,
and $\log([Ca^{2+}]_{\text{in}})$.

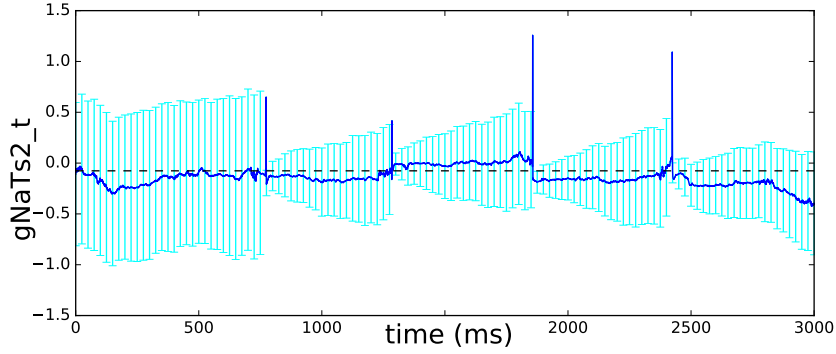


(d) $\log(\bar{g}_{\text{CaHVA}})$, $\log(\bar{g}_{\text{Kv3.1}})$,
 $\log(\bar{g}_{\text{SK}})$, $\log(\bar{g}_{\text{CaLVA}})$, $\log(\bar{g}_{\text{H}})$,
 $\log(\bar{g}_{\text{Nat2}})$, $\log(\bar{g}_{\text{L}})$, $\log(\gamma)$, $\log(\text{decay})$.

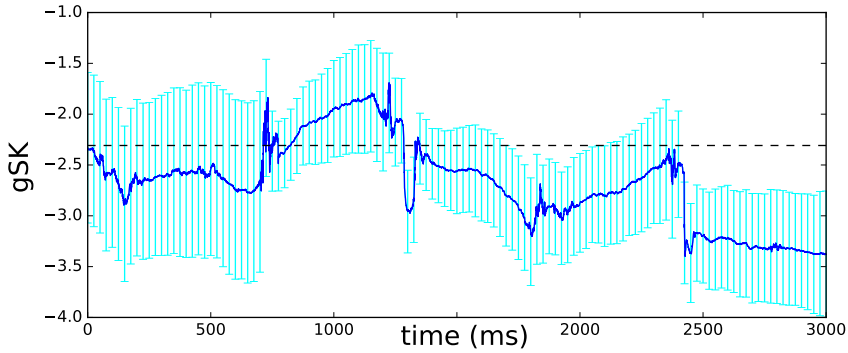
Figure 9.1: Approximated filtering distribution computed by a single run of the EnKF. The dashed black lines mark the true trajectory $x_{0:J}^{\dagger,(\text{step})}$ and the blue solid lines the filtering mean. The cyan bands represent the amplitude of the SD-confidence interval. In the caption of (c) and (d) the variable names are listed row-wise (from the left to the right), and then column-wise (from the top to the bottom).

the correct region is eventually recovered after the spike, but this is not the case for all parameters. For instance, the posterior distribution for parameter \bar{g}_{SK} (depicted in Figure 9.2b) shows that such excursion occasionally drive the filtering distribution away from the correct region.

This could be an issue related to a bad choice of the setting parameters N , σ_v , σ_g . Alternatively, it could even be a consequence of a bad choice of state-space model (σ_g decreasing in time could be a solution) or a bad choice of the parameter-estimation approach (a maximum likelihood estimator for parameters could be employed). In the following explore the former of these possible solutions.



(a)



(b)

Figure 9.2: Focus on two parameters (magnified from Figure 9.1d): posterior distribution for parameter component $\bar{g}_{\text{Nat}2}$ (a) and \bar{g}_{SK} (b), both in a log-scale. The dashed black lines mark the true parameter values, the blue solid lines the filtering mean, and the cyan bands represent the amplitude of the SD-confidence interval for the filtering distribution.

Now, when we execute a single run of the EnKF, we have a time-dependent filtering distribution for both the variable component x and the parameter component θ of the augmented signal variable. For the dynamic component x , this provides a point estimator of the signal at each time step (hidden unobservable variables m_{ion} and h_{ion} included). Note that if the initial condition is badly initialised (unreliable choice for μ_0 and C_0), this estimation is only credible after several filtering steps.

On the other hand, a time-dependent filtering estimation is not desirable for the modelling parameters, since these are static by definition. Then, some trick to get a static estimator $\hat{\theta}$ from the time-evolving filtering distribution is needed. Denoting $(\mu_{x_j}, \mu_{\theta_j})^T = \mathbb{E}[(X_j, \theta_j) | y_{0:j}]$ the approximated filtering

mean at time j , a possible choice is to consider μ_{θ_j} – the mean of the filtering distribution at the end of the data assimilation time window – as an estimator for θ^\dagger . However, Figure 9.2 shows that the filtering distribution can be driven away from a good region right at the end of the data assimilation window. Thus, a more robust approach consists in considering a time-average over a final portion of the data assimilation window, as we do in the twin experiment presented in the previous chapter. In practice, in this experiment we take the average of the filtering θ -mean over the second half of the data assimilation window²

$$\hat{\theta} = \frac{1}{J - (J/2 + 1)} \sum_{j=J/2+1}^J \mu_{\theta_j}, \quad (9.1)$$

which provides a point estimator for the modelling parameter θ^\dagger .

In order to test the estimates we obtain for both hidden variables and modelling parameters, we run a forecast-skill analysis. In particular, for each of the stimuli $I_{\text{ext}}^{(\text{in})}$, $I_{\text{ext}}^{(\text{out1})}$, and $I_{\text{ext}}^{(\text{out2})}$ defined in Section 5.3.3, a new simulation is launched (same solver and same numerical step) using the filtering mean at time $j = J$ as initial condition (i.e., $x(t_0) = \mu_{x_J}$), and parameters given by the $\hat{\theta}$ defined in (9.1). We denote the resulting trajectories $\hat{x}_{J:2J}^{(\text{in})}$, $\hat{x}_{J:2J}^{(\text{out1})}$, and $\hat{x}_{J:2J}^{(\text{out2})}$, respectively.

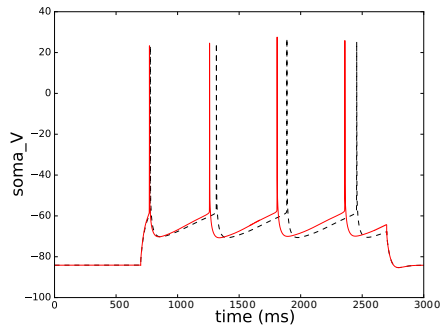
An example of estimated trajectory for the particular run described in this section is given in Figure 9.3. We remark that all estimated trajectories (plain red lines) anticipate the true spike train (dashed black lines) of some millisecond, with a lag which increases in time. For input current $I_{\text{ext}}^{\dagger(\text{out1})}$, there is even an extra spike in the estimated trajectory (Figure 9.3c).

Although the estimation is, all in all, good, the presence of both asynchronous and extra spikes is a clear sign that this run did not produce a satisfactory estimation considering that we are in a twin experiment setting. As a consequence, in the following section we investigate if adjusting the EnKF parameters N , σ_v , and σ_m can be enough to ameliorate the prediction performances. However, we need a metric on the space of estimated trajectories to compare the outcome of different possible adjustments to the DA algorithm. To this aim, we adopt the performance score s defined in (4.13).

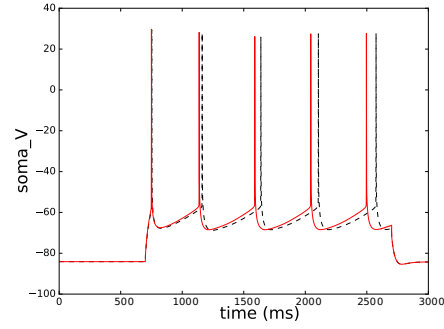
9.1.3 ANOVA tests results

Since the estimation result we obtained for $N = 100$, $\sigma_v = 1\text{e-}2$, and $\sigma_g = 1\text{e-}2$ is not satisfactory, we look for a better collection of setting parameters. In particular, we consider two levels of dynamical noise for the membrane

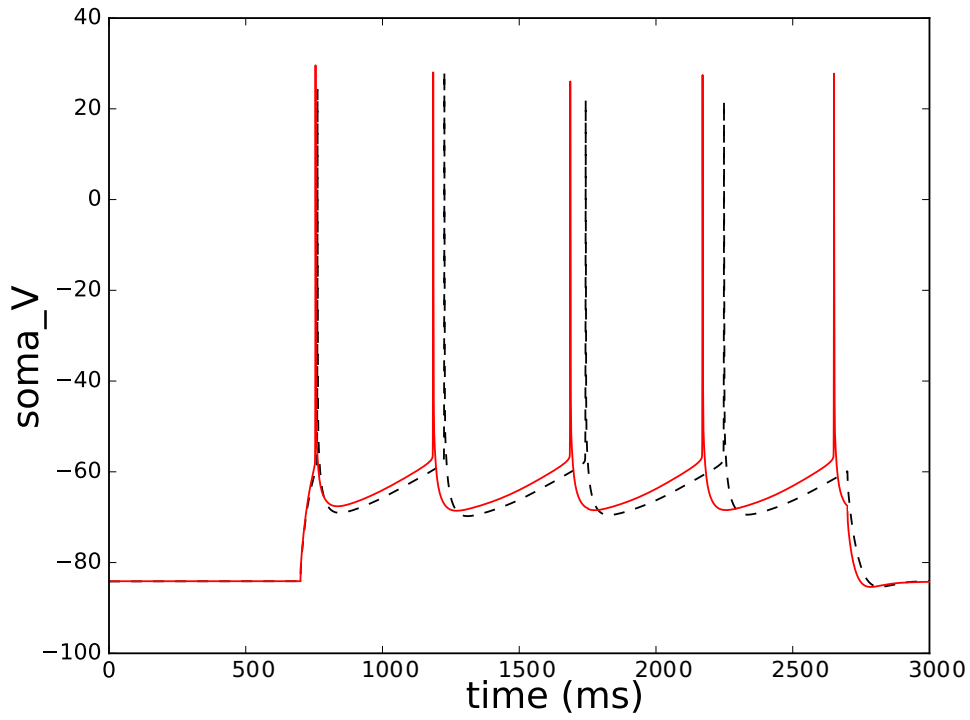
² i.e. over $j \in \{J/2 + 1, \dots, J\}$ corresponding to the interval $t_j \in [1500.25\text{ms}, 3000\text{ms}]$



(a) $\hat{x}_{J:2J}^{(in)}$ (plain red line) versus $x_{J:2J}^{\dagger(in)}$ (dashed black line).



(b) $\hat{x}_{J:2J}^{(out2)}$ (plain red line) versus $x_{J:2J}^{\dagger(out2)}$ (dashed black line).



(c) $\hat{x}_{J:2J}^{(out1)}$ (plain red line) versus $x_{J:2J}^{\dagger(out1)}$ (dashed black line).

Figure 9.3: Signal estimation in the in-sample time window (a), the first out-of-sample time window (b) and the second out-of-sample time window (c).

potential variable V and for the maximal conductances \bar{g}_{ion} ($\sigma_v = 1e-2, 1e-3$ and $\sigma_g = 1e-2, 1e-3$) and five ensemble sizes ($N = 50, 100, 200, 500, 1000$). For

each set of parameters, ten independent runs were executed in order to have a minimal representative statistics for each set of parameters. Then, an analysis of variance (ANOVA) test is performed in order to rank the results obtained from a full-factorial design (i.e. all possible combination of parameters are tested).

First, let us inspect the results of the new setting parameters in a graphical way. Figure 9.4 shows all results obtained, by representing the box-and-whisker plot for the performance score statistics as defined in (4.13). The box plots are grouped for level of the three factors we considered, namely the ensemble size N , the standard deviation for the dynamical noise acting on the membrane potential σ_v , and the standard deviation for the dynamical noise acting on the \bar{g}_{ion} components σ_g .

In particular, Figure 9.4a shows that the best median score of approximately $s = 0.65$ is attained for $N = 100$. The first-third quartile region is similar for both ensemble sizes $N = 100$ and $N = 200$, but the latter appears to be more skewed towards larger values of the performance score (worse estimation). The largest dispersion is obtained for the smallest ensemble size $N = 50$, whereas the distribution dispersion appears to decrease substantially for the largest ensemble sizes $N = 500, 1000$. However, the performance score statistics for these large-ensemble sizes is consistently larger than for smaller ensemble sizes. A few outliers, representing extremely good estimation scores, appear for $N = 50, 100$ and 200 . In conclusion, in accordance to previous findings [76], an ensemble of one hundred particles seems to be the best choice in this example.

As for Figure 9.4b and Figure 9.4c, both graphs show that a larger standard deviation ($\sigma_v = 1e-2$ and $\sigma_g = 1e-2$) produces less variability in the performance scores. However, for both $\sigma_v = 1e-3$ and $\sigma_g = 1e-3$, the distribution appears to be more skewed towards the small performance score region (corresponding to a very good estimation). Nevertheless, in both graphs the median scores appears to be the same for both levels of dynamical noise, suggesting that these parameters do not play a crucial role.

Then, we look for a quantitative confirmation of these results. To this end, we performed a series of ANOVA tests, considering many different statistical models, with and without interactions. Although none of the statistical models we considered turned out to be explanatory enough (very small adjusted R^2 value), in all cases the p -value for the N -effect was smaller than the significance threshold $\alpha = 5\%$, whereas neither the σ_g -effect or the σ_v -effect was significant. This suggests that the effect of the ensemble size N is indeed relevant while, on the contrary, the effect of σ_g or σ_v does not affect the estimation performance with any statistical significance. Also, applying a Tukey's honest significant

difference (HSD) test for the N factor, we verified what groups are actually significantly different. In fact, Figure 9.5 shows that the ensemble size of 100 particles produces estimation performances which are significantly different from the results obtained with an ensemble of either 50, 500, or 1000 particles, respectively. No other differences were detected with this set of results.

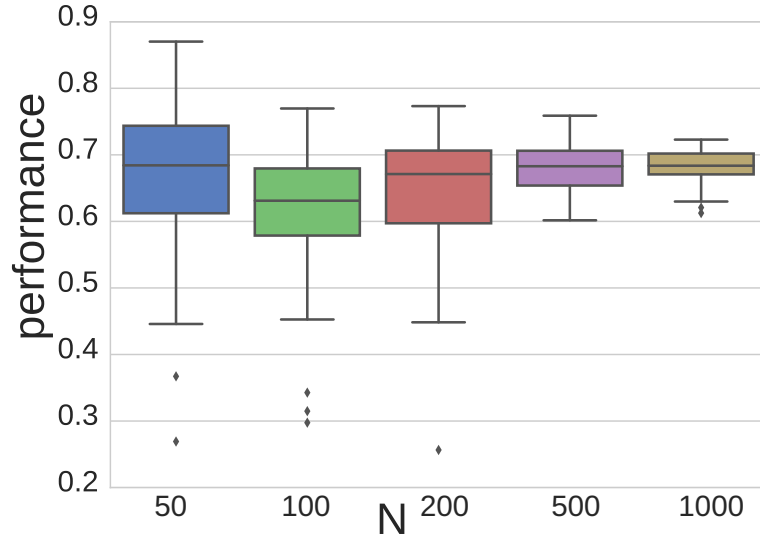
Although we verified the difference is not statistically significant for all factors, we tried to identify the triple (N, σ_v, σ_g) which produces the best performances. To this end, we further investigated the results obtained by plotting the factor plots for all performance scores. In accordance with what noted in the previous figure, Figure 9.6 shows that the smallest performance scores are obtained for $N = 100$, but also that for such ensemble size $\sigma_g = 1e-3$ (left panel) produces substantially smaller confidence intervals.

Besides, both values of σ_v produce similar profiles (especially for $\sigma_g = 1e-3$), even though $\sigma_v = 1e-3$ appears to produce slightly smaller mean performance scores. In conclusion, in this experimental setting, the best triple of setting parameters appears to be $(N, \sigma_v, \sigma_g) = (100, 1e-3, 1e-3)$, even though both graphical inspection and statistical tests suggest that σ_v does not play a crucial role.

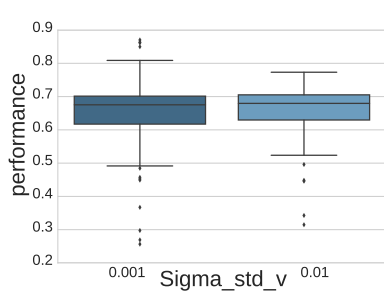
As a concluding remark, note that a larger ensemble size N corresponds to a larger computational load. In fact, increasing tenfold the ensemble size results in a computational load that is over 8 times larger in our experiment, as reported in Table 9.7 and graphically represented in Figure 9.8.

9.2 Second experiment: biased initial condition for parameters

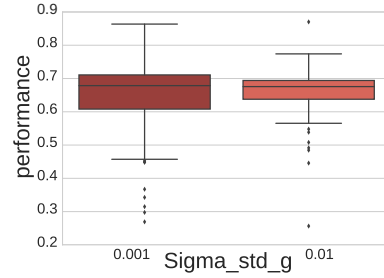
We remark that in the experimental setting described in the previous section, the stochastic initial condition $\mathcal{N}(\mu_0, C_0)$ is centred on the true initial condition and the true parameter, i.e. $(\mu_{x_0}, \mu_{\theta_0}) = (x_0^\dagger, \theta^\dagger)$, where $\mu_0 = (\mu_{x_0}, \mu_{\theta_0})^T$. Since we take the filtering mean as point-estimator of the signal, this means that the method always starts from an initial guess which is very close to the true value. In fact, the i.i.d. ensemble $\{Z_0^{(n)}\}_{n=1}^N \sim \mathcal{N}(\mu_0, C_0)$ will always have an ensemble mean $\sum_{n=1}^N Z_0^{(n)}$ which is very close to the initial mean μ_0 . This is a very simple case which is not realistic in real-world application. In fact in an experimental-data setting, the true initial condition is not known at all, especially the one of the modelling parameters. Also, this means that the parameter approximation is very likely to worsen from such a good initial guess, even when the overall signal estimation is good enough. For instance, compare the distance between the filtering mean (blue line) and the true parameter value



(a) Performance score versus ensemble size N : $N = 50$ (blue), $N = 100$ (green), $N = 200$ (red), $N = 500$ (violet), and $N = 1000$ (ochre).



(b) Performance score versus σ_v , the standard deviation of dynamical noise for the V -component: $\sigma_v = 1e-3$ (dark blue), and $\sigma_v = 1e-2$ (light blue).



(c) Performance score versus σ_g , the standard deviation of the dynamical noise for the maximal conductances (\bar{g}_{ion} 's): $\sigma_g = 1e-3$ (dark red), and $\sigma_g = 1e-2$ (light red).

Figure 9.4: Box-and-whisker plots of the performance scores (defined in (4.13)) obtained among all 100 simulations, grouped by ensemble size N (a), by σ_v (b), and by σ_g (c). The bottom (respectively top) of the coloured boxes mark the first quartile (respectively third). The bar inside the coloured boxes mark the median (i.e. the second quartile), whereas the whiskers extension is of 1.5 times the interquartile range. Individual points outside the whisker region identify the outliers of the distribution.

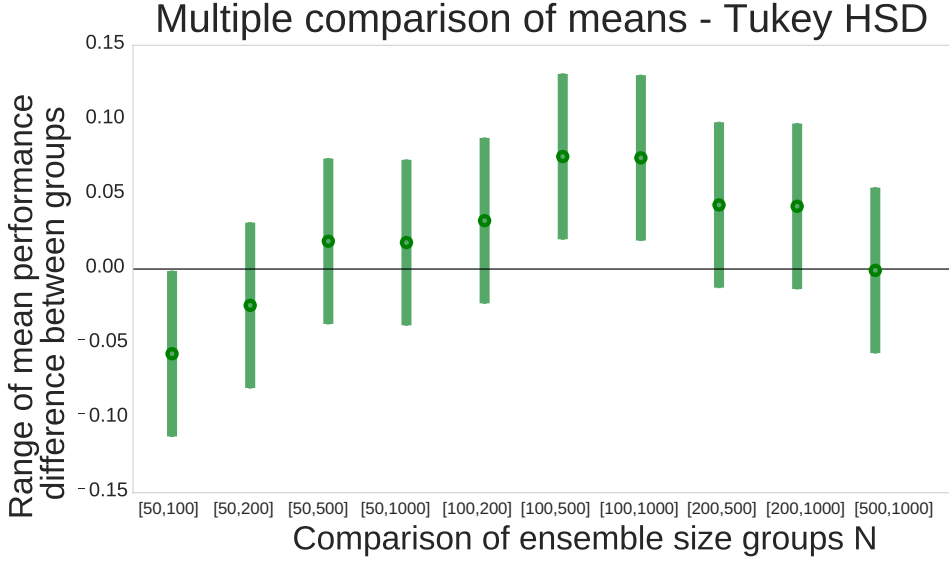


Figure 9.5: Tukey’s honest significance difference (HSD) test for the ensemble size. The dark green points represent the mean difference in the performance score between two groups, whereas the vertical bars represent the confidence interval for the mean. Whenever the confidence interval does not cross the horizontal line $s = 0$, the difference between the two groups is statistically significant.

(black dotted line) at the beginning of the data assimilation time window to the same distance at its end ($t = 3000$ ms) in Figure 9.2b. Hence, we need to address the case of a biased initial condition in order to make steps further in the direction of applying the DA methods described in Chapter 1 to real experimental data.

In what follows we show the result of applying the ensemble Kalman filter as described in the previous section, but assuming the initial parameter mean μ_{θ_0} is a random vector not centred on the true parameter set (i.e. $\mathbb{E}[\mu_{\theta_0}] \neq \theta^\dagger$).

In practice, we assume the initial variables to be still centred around the true initial condition ($\mu_{x_0} = x_0^\dagger$), but the parameter component of the initial mean is assumed to be a r.v. which takes values in a neighbourhood of the true parameter set. In particular, we set

$$\mu_{\theta_0} \sim \mathcal{U}(0, 2r \cdot \theta^\dagger), \quad (9.2)$$

for the untransformed (i.e. positive) parameters, so that its expected value is $\mathbb{E}[\mu_{\theta_0}] = r\theta^\dagger$. Since we chose a unbounded confidence interval for all parameters, we decided to limit the support of the initial mean to a bounded interval of size proportional to the true value. Other distributions on \mathbb{R}^+ could have been employed, but the r introduced in (9.2) allows one to deal with the dif-

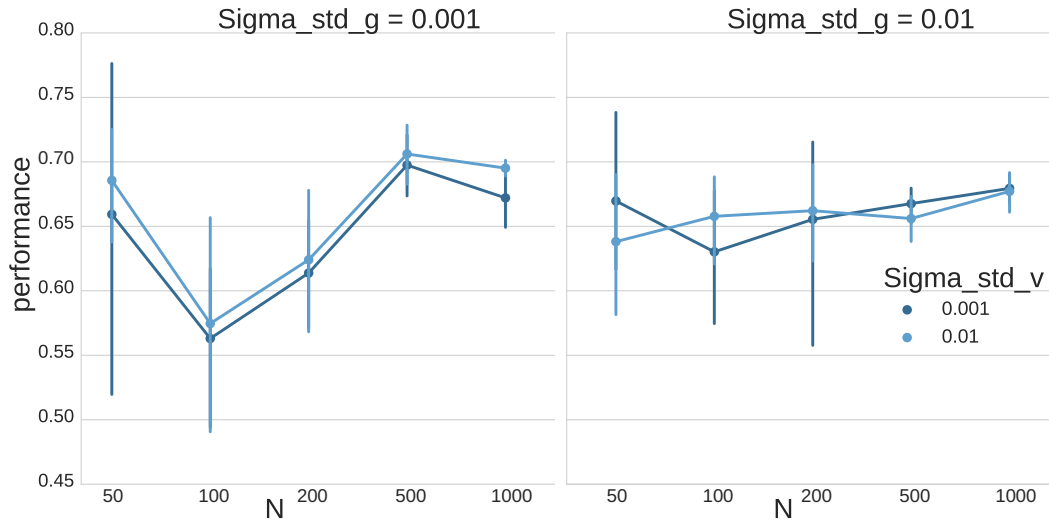


Figure 9.6: Mean performance score and error bars as a function of N , for $\sigma_g = 1e-3$ (left panel) and $\sigma_g = 1e-2$ (right panel). The dark blue lines correspond to $\sigma_v = 1e-3$ and the light blue lines to $\sigma_v = 1e-2$.

ferent scales of the single modelling parameters (the components of θ) while parametrizing the expected value of the initial relative error from the true value.

As a test study, we run 40 instances of the EnKF with $r = 35$, dynamical noise parameters $\sigma_v = \sigma_g = 5e-3$, $\sigma_m = \sigma_\gamma = 1e-2$, $\sigma_z = \sigma_{[Ca^{2+}]_{in}} = 1e-4$, $\sigma_{decay} = 1e-3$ and initial condition as described in Section 9.1.1. In addition, the measurement noise standard deviation is reduced to $\Gamma^{1/2} = 0.3$ mV. Finally, note that in this section, we employ the base-performance score given by the SPIKE-distance D_S (see Chapter 7) instead of the $d^{(rel)}$ -distance proposed in Section 9.1.

9.2.1 Signal estimation results

A brief summary of the signal estimation performance statistics is given in Table 9.9, which lists the mean, the standard deviation and the coefficient of variation³ across the 40 runs of the D_S -distance between the estimated membrane potential trace and the true spike train in:

- **in** – the in-sample time window (first row);
- **out1** – the first out-of-sample time window (second row);

³The coefficient of variation of a random variable with expected value μ and standard deviation σ is defined as $cv = \sigma/|\mu|$

wall time	mean	sd
$N = 50$	0.13	0.01
$N = 100$	0.43	0.03
$N = 200$	0.51	0.11
$N = 500$	1.81	0.33
$N = 1000$	3.61	0.62

Table 9.7: Mean and standard deviation of the computational wall time (in hours) for different ensemble sizes.

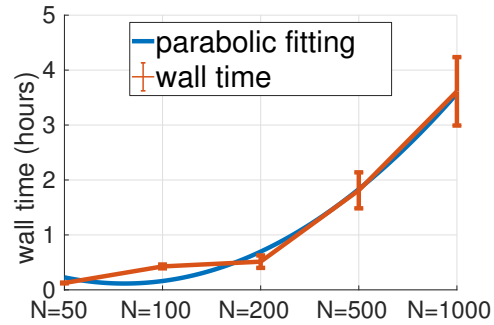


Figure 9.8: Graphical representation of the computational wall time as a function of the ensemble size N .

- **out2** – the second out-of-sample time window (third row).

We note that these three average values are consistent in all three windows, with a mean D_S -error approximately valued 0.210, an average standard deviation of about 0.120, and a coefficient of variation 0.570. As a consequence, we expect the signal estimation quality to be similar in both the in-sample and the out-of-sample time windows.

The last two columns shows the D_S -score for two notable runs which we later use as a benchmark to assess the actual estimation quality:

- 1) the **best run**, which we define as the run for which the sum of the D_S -distance in the three time-windows is minimal;
- 2) the run which produced the in-sample performance score closest to the mean value. We name such run the **mean- D_S run**.

We also present a more explanatory representation of the performance distribution in Figure 9.10. The figure shows a scatter plot of the bivariate dataset formed by the performance score in the in-sample time window (horizontal axis) and the average out-of-sample performance score $\frac{1}{2}[D_S(\text{out1}) + D_S(\text{out2})]$, along with the contour plot of the estimated joint p.d.f.. First, we note that the high positive correlation ($\rho = 0.93$) suggests that a good in-sample performance is a reasonable predictor for obtaining a good out-of-sample estimation: if the in-sample potential trace is tracked down accurately, it is likely that the out-of-sample prediction will be accurate too. Note that this is not the case for the D_S -performance score of the filtering mean, which is typically very good independently of the actual estimation quality. In fact, Pearson's correlation coefficient between the D_S -performance of the filtering mean and the average out-of-sample performance is significantly lower ($\rho = 0.42$).

D_S	mean	sd	cv	best	mean- D_S
in	0.211	0.119	0.567	0.049	0.202
out1	0.209	0.128	0.610	0.016	0.141
out2	0.209	0.112	0.534	0.009	0.261

Table 9.9: Mean, standard deviation, and coefficient of variation for the voltage-estimation performance across 40 runs of the twin-experiment with biased initial parameter distribution (9.2) for $r = 35$ in all time windows. The last two columns list the performance score for the best run and the run which produced the in-sample performance score closest to the mean.

In addition, we note that the estimated joint p.d.f. has two peaks. The first one is an absolute maximum located around the point of planar coordinates $D_S(\text{in}) = \frac{1}{2}[D_S(\text{out1}) + D_S(\text{out2})] \approx 0.27$ (i.e. performance scores slightly larger than the mean). On the other hand, the second maximum corresponds to much better performance scores (the peak in the lower-left corner of Figure 9.10). Note that the value corresponding to the best run lies in the neighbourhood of the second peak (red dot in the figure). Also, the value for the run which produced an in-sample performance in the mean range ($D_S(\text{in}) \approx 0.211$ as listed in Table 9.9) is highlighted in orange.

Now, in order to better understand what these values correspond to in estimation performance terms, we present in Figure 9.11 the raster plot of the filtering mean, the in-sample, and the two out-of-sample estimates for both the best and the mean- D_S runs. Although estimation resulting from runs in the higher-peak region essentially does not fit the data (in the best case scenario, only the first spike is predicted in every time window in these runs – not shown in the picture), the raster plot in Figure 9.11a shows that the performance for the best run is essentially perfect. In addition, the raster plot for all other runs with performance score in the lower-peak region (the **second-best runs**) are very close to the one of the best result. In fact, eight out of ten second-best runs have the correct number of spikes – although their timing is not as precise as the almost perfect match of the best run (see Figure 9.16). As for the run in the mean range, the estimation output is reasonable but far from providing a sufficient matching of the assimilated data (see Figure 9.11b).

In conclusion, we remark that the overall performance distribution is centred around values that do not predict out-of-sample response appropriately. Nonetheless, the presence of a peak in the performance distribution and a whole bunch of good runs in the best-run vicinity suggests that the EnKF can be effective in the biased initial parameter regime too, but only if focusing on those few runs which produce the smallest prediction errors. Note that this

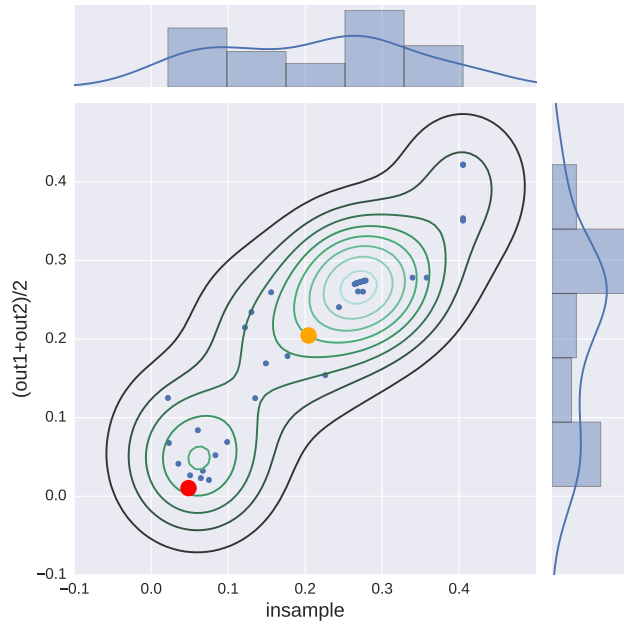
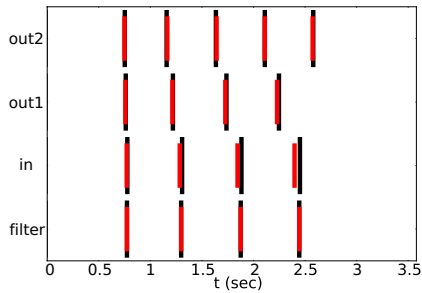
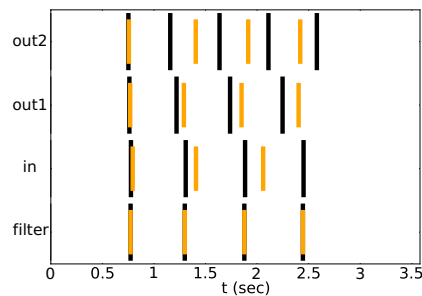


Figure 9.10: Scatter plot and estimated joint p.d.f. of the D_S -performance score in the in-sample time window (x -axis) and the average performance in the two out-of-sample time windows (y -axis), along with marginal histograms and corresponding univariate kernel density estimates. The value corresponding to the best run is marked in red (see the raster plot in Figure 9.11a) and the mean- D_S run in orange (Figure 9.11b).



(a) Raster plot corresponding to the best run.



(b) Raster plot for the mean- D_S run.

Figure 9.11: Raster plot of the filtering mean [`filter`], the estimated trajectory in the in-sample [`in`], the first out-of-sample [`out1`], and the second out-of-sample time window [`out2`] for: (a) the best run; (b) the mean- D_S run.

procedure is similar to the common practice, in the machine-learning community, to select the successful runs if the training procedure by testing the prediction on some validation data, which are different from both the training set and the test set, as we detail in Section 9.2.4.

9.2.2 Parameter estimation results

Up to now we only analysed the signal estimation performances in terms of the predicted voltage trace. But what about the parameter estimation performance? Before proceeding in this direction, we remember that in case of parameters we are entitled to consider errors computed on both the logarithmically transformed parameters or on the untransformed physical parameters. (In the case of signal estimation performances no transformed variable had to be taken into account because only the estimation of the unbounded membrane potential is considered).

Only looking at the mean and standard deviation of the log-transformed parameter relative error in Table 9.12a it looks like the average parameter estimation output do not provide reliable results. In fact, some mean errors appear significantly large (for instance, $|\log(\hat{g}_{\text{Nat}2}) - \log(g_{\text{Nat}2}^\dagger)|/|\log(g_{\text{Nat}2}^\dagger)| \approx 9$). In addition, some of those parameters that do not have a very large relative error, have a relatively large variability (see $\bar{g}_{\text{Kv}3.1}$ for instance). Overall, Table 9.12a seems to suggest that the typical output of the EnKF will not be of much use in terms of parameters estimation.

However, what about the relative error for the physical counterpart of the parameters? Does the situation change when these are turned into their physical version? Well, the situation does change but not in a positive sense. Indeed, turning to the real scale unveils the true amplitude of the typical relative errors. Table 9.12b shows that such errors inflate of several orders of magnitude in the physical scale, even for those parameter that have a small relative error in their log-version. In addition, we remark that the error range is very variable across different parameters. For instance, $\bar{g}_{\text{Kv}3.1}$, \bar{g}_{SK} , and decay have mean relative errors larger than 10^8 in the physical scale, whereas \bar{g}_{CaHVA} and $\bar{g}_{\text{Nat}2}$ have mean errors smaller than 10^1 . As a consequence, the average value presented in the last row of the table does not really capture the overall performance across different parameters, because it is dominated by the large ones. Only the coefficient of variation keeps having similar orders of magnitude for all parameters also in the physical scale. However, the fact that all the coefficients of variation are larger than two demonstrates a very large variability in the parameter estimation performances.

In conclusion, we can draw three messages from Table 9.12:

$ \frac{\log \hat{\theta} - \log \theta^\dagger}{\log \theta^\dagger} $	mean	sd	cv	$ \frac{\hat{\theta} - \theta^\dagger}{\theta^\dagger} $	mean	sd	cv
\bar{g}_{Nat2}	9.29	3.62e1	3.90	\bar{g}_{Nat2}	8.78	5.39e1	6.14
$\bar{g}_{\text{Kv3.1}}$	5.80e-1	2.12	3.68	$\bar{g}_{\text{Kv3.1}}$	5.15e8	3.22e9	6.24
\bar{g}_{SK}	1.37	2.02	1.48	\bar{g}_{SK}	4.06e9	2.53e10	6.24
\bar{g}_{CaHVA}	2.40e-1	4.70e-1	1.95	\bar{g}_{CaHVA}	2.53	7.77	3.07
\bar{g}_{CaLVA}	5.80e-1	4.40e-1	0.77	\bar{g}_{CaLVA}	8.18e1	2.16e2	2.64
γ	3.20e-1	3.30e-1	1.02	γ	3.64e2	1.58e3	4.34
decay	1.03	1.25	1.22	decay	2.07e12	1.27e13	6.14
Average	1.91	6.12	2.00	Average	2.96e11	4.86e12	4.98

(a) Transformed case (logarithmic scale). (b) Untransformed case (physical parameters).

Table 9.12: Mean, standard deviation (sd) and coefficient of variation (cv) of parameter estimation relative error of the log-scaled parameters (a) and the untransformed physical parameters (b).

- i)* looking at the parameter estimation errors in the transformed version (in this case logarithmic) is misleading, especially in evaluating the relative estimation performance across different parameters;
- ii)* the mean performance of the EnKF in the physical scale is not good in terms of parameter estimation (and actually extremely bad for many parameters)
- iii)* the parameter estimation variability is significantly large, not only in absolute sense but also when compared to the mean value.

As for the prediction results, what we found is that the typical outputs of the EnKF (i.e. those in the (mean \pm SD)-range) are not reliable in terms of parameter estimation. Does this also mean that the parameter estimation is always unreliable? In order to answer this question we need to identify and explore the output of some relevant runs.

Instead of identifying the best runs relying on the parameter errors statistics (e.g. inspecting the run with the minimum average parameter error), we decide to base our investigation on the D_S -performance score for the signal estimation, as presented in the previous section. Such choice is motivated by the fact that in an experimental setting we are not aware of the true parameters set. Hence, any kind of parameter error is incalculable when dealing with experimental data. On the other hand, exploring the in-sample (or out-of-sample) signal prediction distribution is possible even when no information on the true parameters is available. As a consequence, we decide to check the

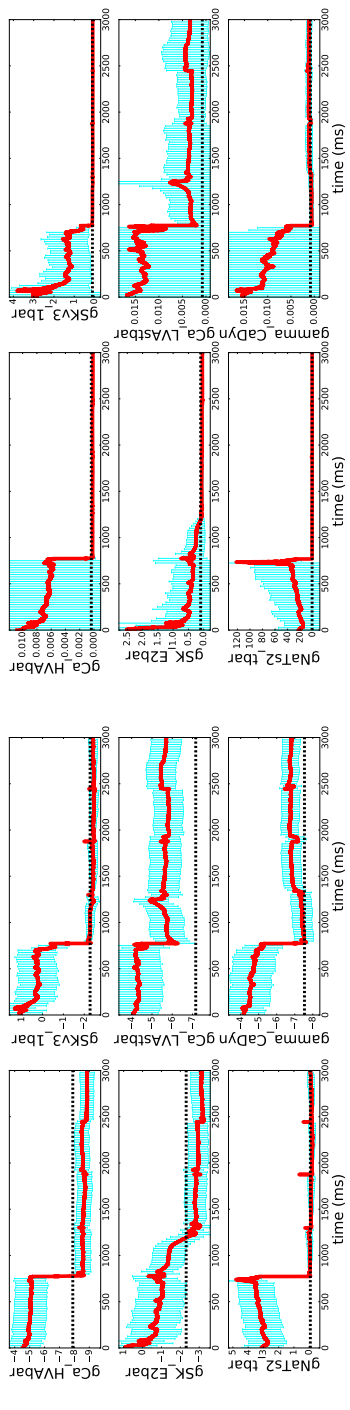
parameter estimation performance for the two runs identified in the preceding section: the best run (marked in red in Figure 9.10) and the run which produced a signal estimation error in the mean D_S -range (orange raster plot in Figure 9.11a).

As shown in Figure 9.13, in both runs the filtering mean does approach the true value (dashed black line) despite the initial mean is much larger. This is true in the logarithmic scale (left column), but even more in the physical scale, i.e. the one that actually matters. Indeed, even those parameters whose final values do not exactly match the true value in the logarithmic scale (e.g. \bar{g}_{SK} , \bar{g}_{CaLVA} and γ) actually get very close to the true value in the physical scale (right column). The only exceptions are parameters γ and decay (not shown) in the mean- D_S run. We report that the other second-best runs (i.e. the other runs with D_S -score located in the region of the lower-left corner peak of Figure 9.10) produce qualitatively similar filtering profiles. In such runs parameters \bar{g}_{CaLVA} and γ are the only ones that occasionally do not track down the true parameter value (not shown here).

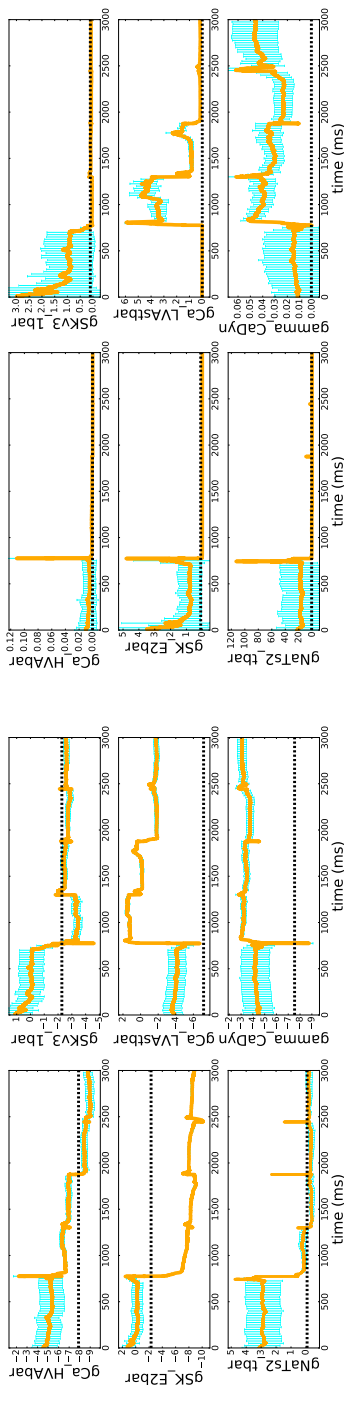
In order to quantify what Figure 9.13 shows only qualitatively, the parameter relative errors for the best run and for the mean- D_S run are presented in Table 9.14. First, we note that the best run has parameter relative errors significantly smaller than the corresponding mean values (compare Table 9.14 and Table 9.12). Indeed, in the mean relative error is at least twice as large as the best run log-transformed parameter error. But the difference is even more clear in the physical scale, where the ratio is of at least one order of magnitude (and up to twelve orders for parameter decay!). However, this is likely to be due to the extremely large errors of some few bad runs which move the mean error towards very large values.

The situation is partially similar for the mean- D_S run, which has relative errors in the physical scale slightly larger than the best run for most parameters. However, the presence of a few very badly estimated parameters (note the large errors in the rows corresponding to \bar{g}_{CaLVA} , γ , and decay in Table 9.14b) is likely to be the cause the significantly different signal-estimation performances illustrated in Figure 9.11. Once again, we remark that the corresponding values in the log-scale table do not give any insight in this sense. As a consequence, we argue that it should not be taken into account in future references: only the physical scale is meaningful and it is hence the only one that should be employed to evaluate parameter estimation quality.

In conclusion, as for the signal-estimation performance distribution, although the typical parameter estimation performance is not good at all, focusing on the best results allows one to obtain reasonable parameter estimates. However, since not all parameter errors are small even in the best run, we argue



(a) Filtering distribution in the log-scale for the best run.



(b) Filtering distribution in the physical scale for the best run.

Figure 9.13: Parameter components of the posterior confidence interval (mean \pm SD, plotted in cyan) and of the filtering mean for: (a)-(b) the best run (filtering mean plotted in red), and (c)-(d) the mean- D_S run (mean plotted in orange). The black dashed lines marks the value of the true parameters. The left column corresponds to the logarithm of the estimated/true parameters, whereas the right column is in the actual physical scale. Note that the component corresponding to modeling parameter decay is not presented here.

$\left \frac{\log \hat{\theta} - \log \theta^\dagger}{\log \theta^\dagger} \right $	best	mean- D_S
\bar{g}_{Nat2}	1.03	2.25
$\bar{g}_{\text{Kv3.1}}$	7.29e-2	1.69e-1
\bar{g}_{SK}	3.37e-1	2.69
\bar{g}_{CaHVA}	1.09e-1	1.16e-1
\bar{g}_{CaLVA}	2.15e-1	7.62e-1
γ	1.02e-1	5.68e-1
decay	7.38e-2	1.36
Average	2.77e-1	1.13

(a) log-transformed parameters

$\left \frac{\hat{\theta} - \theta^\dagger}{\theta^\dagger} \right $	best	mean- D_S
\bar{g}_{Nat2}	7.55e-2	1.57e-1
$\bar{g}_{\text{Kv3.1}}$	1.53e-1	3.20e-1
\bar{g}_{SK}	5.41e-1	9.98e-1
\bar{g}_{CaHVA}	5.77e-1	5.97e-1
\bar{g}_{CaLVA}	3.67	2.33e2
γ	1.16	7.11e1
decay	5.38e-1	2.76e3
Average	9.59e-1	4.39e2

(b) Untransformed case (physical parameters)

Table 9.14: Relative errors for both the best run and the mean- D_S run (marked in red and orange, respectively, in Figure 9.10) for all estimated parameters computed on: (a) the logarithmically transformed parameters, and (b) the untransformed physical parameters.

that one should take with care the parameter estimates the EnKF provides.

9.2.3 Improvement rate of the parameter estimation

Lastly, let us discuss the parameter estimation improvement rate: is the parameter estimate $\hat{\theta}$ defined in (9.1) better than the initial guess μ_{θ_0} ? In case it is, can we quantify how much the estimation improve? To answer these questions we consider the **improvement rate** defined as

$$\left| \frac{\mu_{\theta_0} - \theta^\dagger}{\hat{\theta} - \theta^\dagger} \right|,$$

i.e. the absolute error of the initial mean divided by the final estimate error (no log-transformed scale is considered in this case). When this value is larger than one it means that the estimation is indeed better than the initial guess. On the other hand, an improvement rate smaller than one corresponds to an estimate worse than the initial guess.

Starting from such observation, we remark that in Table 9.15 all entries of the best run are larger than one, and that the average improvement factor in this case is approximately of two order of magnitudes. The improvement factor for the mean- D_S run is often larger than one too, but the estimates for \bar{g}_{CaHVA} , γ , and decay are worse than the initial guesses. This means that not only the parameter estimate improves in the best run, but often also in the run falling the expected-value range as well. However, we note that for the

$\frac{ \mu_{\theta_0} - \theta^\dagger }{\bar{\theta} - \theta^\dagger}$	best	mean-D_S
\bar{g}_{Nat2}	3.08e2	1.42e2
$\bar{g}_{\text{Kv3.1}}$	1.87e2	8.10e1
\bar{g}_{SK}	4.44e1	2.78e1
\bar{g}_{CaHVA}	4.84e1	4.26e1
\bar{g}_{CaLVA}	5.60	1.16e-1
γ	2.41e1	3.01e-1
decay	4.02e1	9.21e-3
Average	9.40e1	4.20e1

Table 9.15: Improvement rate for all estimated parameters in the best run and in the mean- D_S -range run.

mean- D_S run even a few wrong parameter estimates can impair any accurate prediction.

9.2.4 Validation in the first out-of-sample time window

In the previous sections, we remarked how a generic run of the EnKF does not provide reliable estimates of the model parameters nor allows the model to predict out-of-sample response to new stimuli. However, we also remarked how selecting those runs producing a small insample and prediction error (those we called second-best runs) allows a reasonable tuning of the model. Such an approach can be related to the common practice, in the machine-learning community, to assess the ability of a given predictive model which has been trained on a **training set** by first checking its performance on a **validation set** which is not part of the training set to avoid overfitting, so to select the best-performing predictive model. Only in a second moment, the selected model is tested on the **test set** and the resulting error is considered as an estimator of the true error (see, for instance, [187, Chapter 11]). Following up on this parallel, in our dataset we may consider **out1** as the validation set used to identify the good runs, and **out2** as the test set. In particular, if we select the **good runs** to be those for which $D_S(\text{out1}) < 0.05$, we obtain a total of 8 successful runs (essentially those in the lower-peak region in Figure 9.10) for which there is a high-quality prediction in the testing time window ($D_S(\text{out2}) \approx 0.04 \pm 0.03$). The **out2**-response is also reported graphically in Figure 9.16. Again, note that since we select the good runs only using potential traces and no parameter values, such selection is possible not only in a twin experiment setting, but also when using real-world data.

However, as far as the model parameters are concerned, we have that the

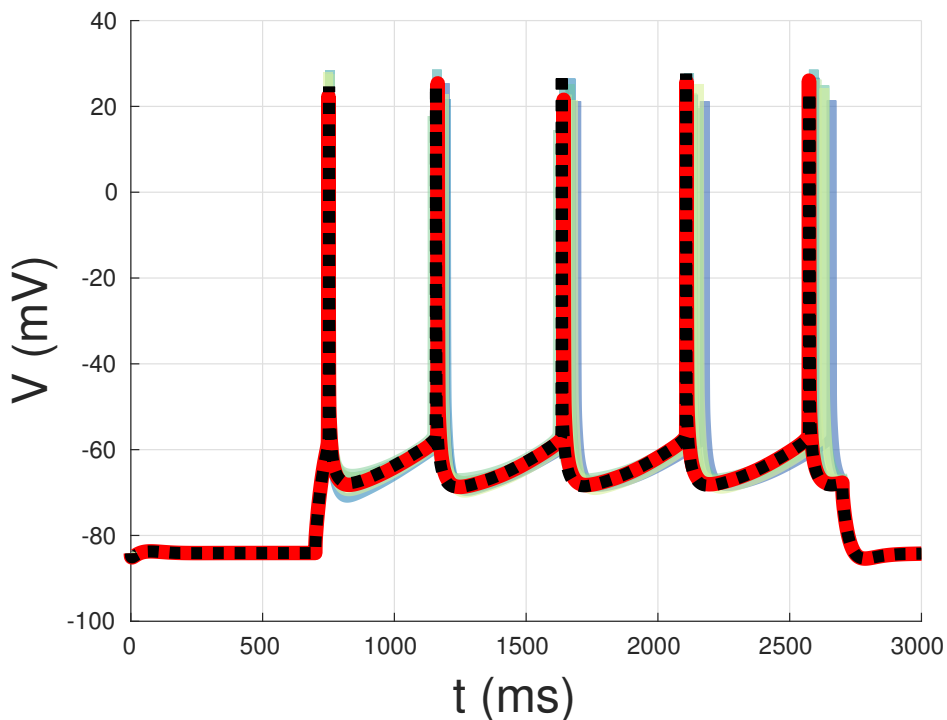


Figure 9.16: Response of the fitted model in the second out-of-sample time window `out2` for the “good runs” selected in the validation step, namely those EnKF runs that produce a small validation error in the first out-of-sample time window ($D_S(\text{out1}) < 0.05$).

estimation quality is rather poor even if we only consider these “good runs” (in a forecast-skill sense). In particular, analysing the mean relative errors reported in Table 9.17a, it appears that only \bar{g}_{Nat2} and $\bar{g}_{\text{Kv3.1}}$ are estimated with sufficient quality, whereas all other parameters have relative error close to 1 or much larger. Nevertheless, note that the parameters with the smallest mean errors are also those whose estimates have minimal coefficient of variation (Table 9.17c), which is a quantity that can give some insight also when true parameter values are not known in advance. Another positive note is that the mean improvement rate is quite large for all parameters (see Table 9.17b).

9.3 Discussion and conclusions

In this chapter, we took the BBP model `L23_PC_cADpyr229_1` into consideration and performed two twin experiments with the idea of progressively approaching truly experimental conditions. In both cases, we applied the en-

$ \frac{\hat{\theta}-\theta^\dagger}{\theta^\dagger} $	mean	sd	$ \frac{\mu_{\theta_0}-\theta^\dagger}{\hat{\theta}-\theta^\dagger} $	$\hat{\theta}$ -mean	$\hat{\theta}$ -cv	θ^\dagger
\bar{g}_{Nat2}	9.72e-2	1.01e-1	1.74e3	8.45e-1	1.19e-1	9.27e-1
$\bar{g}_{\text{Kv3.1}}$	1.62e-1	1.19e-1	2.23e2	8.84e-2	1.70e-1	1.03e-1
\bar{g}_{SK}	2.79	5.25	2.91e1	2.36e-1	2.43	9.94e-2
\bar{g}_{CaHVA}	9.09e-1	6.17e-1	1.05e2	5.14e-4	7.51e-1	3.73e-4
\bar{g}_{CaLVA}	1.45e1	1.50e1	6.78	1.19e-2	1.00	7.78e-4
γ	1.26e3	3.08e3	4.22e1	6.73e-1	2.44	5.33e-4
decay	8.76e2	2.26e3	1.83e1	3.00e5	2.58	3.43e2

(a) Relative error. (b) Mean improvement rate. (c) Estimated values. (d) True values.

Table 9.17: Parameter estimation quality for the “good runs” selected in the validation step, namely those EnKF runs that produce a small validation error in the first out-of-sample time window ($D_S(\text{out1}) < 0.05$)

semble Kalman filter to an augmented state-space model, so to estimate the model parameters in addition to the unmeasured activation variables and the internal calcium concentration. The transformation approach described in Section 4.3 was adopted to deal with the boundedness of state variables and model parameters. Enforcing such bonds was necessary because of implementation issues, but also to guarantee the interpretability of the parameter estimation results.

In the first twin experiment, the objective was to fine tune the EnKF parameters (namely, the ensemble size N , and the dynamical noise variances for the activation variables σ_m^2 and for the maximal conductances σ_g^2) in order to assess the most effective configuration which allows the model to fit the data and predict out-of-sample responses to new stimuli, while confining the computational cost if possible. In this preparatory test, we imposed initial conditions centred on the true values of the model parameters (unbiased initial condition). Our results suggest that an ensemble of relatively small size (composed of a total of $N = 100$ particles) performs significantly better than larger ensembles in predicting the neuron behaviour. Since previous applications of the EnKF to a hydrodynamic models obtained good estimations for a similar ensemble size and found no gain in further increasing the ensemble size [76], this appears to be a relatively robust property of the EnKF. In addition, note that other studies advocate such general belief in the practitioners community to motivate their effort aimed at theoretically justify the effectiveness of the EnKF for small ensemble sizes [114, 184]. As a desirable side effect, we found that an ensemble of 100 particles results in a computational wall time of

about $25 \text{ min } 35 \text{ s} \pm 2 \text{ min } 4 \text{ s}$, which reduces 4.2 times the computational load with respect to a ($N = 500$)-sized EnKF ($1 \text{ h } 48 \text{ min } 41 \text{ s} \pm 19 \text{ min } 38 \text{ s}$), and 8.4 times the average computational load of a ($N = 1000$)-sized EnKF run ($3 \text{ h } 36 \text{ min } 49 \text{ s} \pm 37 \text{ min } 21 \text{ s}$). As far as dynamical noise variances σ_v and σ_g are concerned, our results suggest that their value do not play any major role, at least in the range we tested (10^{-2} , 10^{-3}), but we found no confirmation of such result in the literature and we suggest care in generalising such result to other models.

Then, in the second twin experiment we made the initial condition for the model parameter unbiased. In particular, we turned μ_{θ_0} into a hyperparameter whose components are, in average, thirty-five times larger than the true parameter values. In such way, the random walk of the parameters in the Θ space is not *pro forma*, and the EnKF does need to explore the parameter search-space in order to retrieve the correct parameter values. In this unbiased case, our findings showed that:

- i)* there is a high correlation between the in-sample and the out-of-sample D_S -performance score, so that a very small in-sample error can be considered as a reasonable indicator that the out-of-sample prediction will be good;
- ii)* the performance score distribution is bimodal, with the lower peak corresponding to essentially perfect in-sample estimations and rather good out-of-sample predictions;
- iii)* one should only focus on the few best runs (measured according to the forecast skill in a validation time window which is not part of the training dataset) when interested in foretelling the neuron behaviour in unobserved conditions, because mean results are not satisfactory;
- iv)* small parameter errors do not guarantee good out-of-sample predictions, so that we should scale down our expectations on the ability of augmented state-space filters to consistently estimating model parameters. Possibly, only few maximal conductances can be tracked down with sufficient accuracy.

Finally, we report here that the average computational wall time for one of the 40 EnKF runs in this biased twin experiment is $8 \text{ min } 37 \text{ s} \pm 6 \text{ s}$, which is significantly smaller than the one obtained using an ensemble of $N = 100$ particles in the previous twin experiment. The reason of such discrepancy was not investigated, but a possible explanation is that a different amount of CPUs was actually employed in the two experiments due to the shared usage of the computing facility by other users.

In conclusion, in light of the last twin experiments, we have more modest hopes in the potentiality of the augmented state-space EnKF. In particular, one cannot expect that all model parameters can be estimated with high accuracy. However, we are still confident that the method is able to provide good predictions in relatively cheap execution time, although it is foreseeable that only a few high-quality runs will provide reasonable prediction results.

In the next chapter, we move to a truly experimental setting where the neuronal data to be assimilated are not generated by the prior model, but they are recorded by experimentalists in a laboratory.

Chapter 10

Assimilating experimental data in a realistic BBP model

In the previous chapters, we applied some Bayesian inference methods to a couple of single-neuron models and showed that we are able to recover the dynamics of the unmeasured variables as well as the values of many modelling parameters in different twin experiments. We drew the conclusion that these methods can be effective, at least when the data are generated by the same model that is then used for the data assimilation. However, this is not of much use for practical purposes. In fact, in the real world one only has experimental data and a proposed model which is not guaranteed to be suitable to reproduce those data. Would the Bayesian methods employed so far be still effective?

In this chapter we address such problem by focusing on the application of some inference methods, but only in the case of data obtained from experiments. As we show, many practical issues arise when dealing with such data.

In the first Section 10.1 we present the experimental dataset concerning pyramidal cells which is used in this work, while in Section 10.2 we introduce and propose a solution to some practical issues. Note that after several preliminary attempts, we noted that the previously used model parameters bounds prove not to allow any reasonable estimation when assimilating experimental data on BBP neuron model L23_PC_cADpyr229_1. As a consequence, in Section 10.2 we describe our proposal for reducing the parameter search space size. Disappointingly, not even such adjustment made the EnKF effective. So, in Section 10.3 we describe our choice for another suitable data-assimilation method. In conclusion, in Section 10.4 we present the data assimilation results which finally gave some acceptable prediction and then draw our conclusions.

10.1 Blue Brain Project experimental dataset

The experimental dataset we consider in this chapter contains the membrane potential traces which characterize the electrophysiology of some layer-5 pyramidal cells of Wistar rats. Such traces were recorded *in vitro* according to several different experimental protocols. Note that this is the dataset employed by the Blue Brain Project to fit the pyramidal cell models for all layers of the neocortical microcircuitry described in [149]. We refer to this paper and its Supplementary material for further details about the experimental protocol conditions.

Unlike our approach, the BBP fine-tuning method is not Bayesian but it employs a multi-objective evolutionary algorithm in order to fit the statistics (namely, the mean and the standard deviation) of some relevant electrophysiological features such as the resting potential, the spike rate, the action potential height, etc. Note that such procedure does not fit any single trace, but uses the same extracted values obtained recording different principal neurons to evaluate the model parameters of all pyramidal cells in the microcircuit. The exact procedure is described in [60] and, more recently, in [203] where the issue of features extraction is addressed as well. In [85], the resulting algorithm is applied in detail in order to generate realistic models of layer 5b pyramidal cells. We acknowledge the Neuroinformatics division of the Blue Brain, directed by Sean Hill at the École Polytechnique Fédérale de Lausanne (EPFL), for kindly providing us all the neurophysiology data mentioned and used in this chapter. These data should be available on the NMC portal soon.

To give an idea of the traces contained in the dataset, a brief overview of the various protocols is given in Figure 10.1, where the activity of a same cell in response to different stimuli is depicted. First, Figure 10.1a exemplifies the **IDRest** protocol, which consists in presenting a step input current of the form (5.15): a negative hyperpolarizing current is injected in the cell for three seconds in order to keep the membrane potential hovering around its voltage base, and then a depolarizing current lasting two seconds is superimposed after the first 700 ms in order to elicit some kind of response. On the other hand, the ramp current protocol **APThreshold** results from considering an input current which increases linearly from zero ampere up to a pre-decided value as in Figure 10.1b. These are two experimental protocols which are commonly known in the electro-neurophysiology community as they allow one to identify some key properties of *in vitro* neurons. However, more exotic protocols are available in the BBP dataset. For instance, the **SineSpec** protocol consists in presenting a sinusoidal input current as the one pictured in the bottom panel of Figure 10.1c, while the **NoiseSpiking** protocol results from injecting an oscillating input current with a stochastic white noise overlapped (see

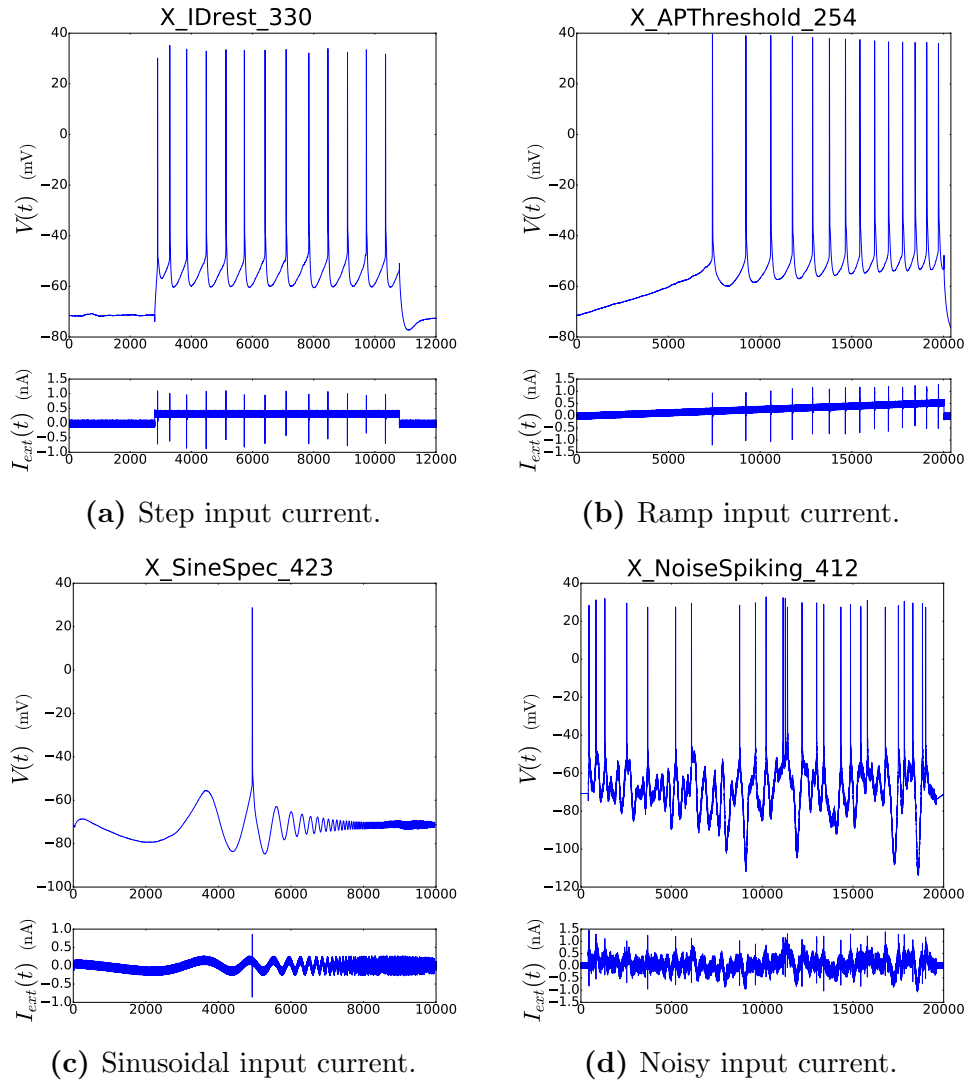


Figure 10.1: Examples of external current profiles and corresponding membrane potential traces for four different experimental protocols considered in the Blue Brain Project dataset for pyramidal cells. In each subfigure, the bottom panel depicts the input current profile whereas the top panel shows the corresponding membrane potential time course. All traces in the picture refer to the same layer 5 pyramidal cell (named C060109A1 in the dataset).

Figure 10.1d).

Notice that in all panels of Figure 10.1 the x -axis (corresponding to the time variable), has different time resolution in each protocol. For instance, the `IDRest` protocol spans a 3000 ms time window with 12 000 points (i.e. four points per millisecond) whereas the `APThreshold` protocol only spans 2100 ms with almost the double of total amount of points (ten points per millisecond). Also note that in general, in correspondence of the membrane action potentials, the underlying current profile has similar spikes. This is due to the rapid membrane potential sign switch which influences the recording device.

10.1.1 Experimental traces selected for data assimilation

As for the simulated data we used in the twin experiments described in the previous chapter, we here only consider data from the step current protocol `IDRest`. In particular, we consider the trace represented in the top panel of Figure 10.1a as the true data $y_{0:J}$, where $J = 19\,999$. As in the simulated data generation process described in Section 5.3.3, the input current is a step current of the form (5.15),

$$I_{\text{ext}}^{(\text{in})}(t) = \begin{cases} I_{\text{hyp}} & t \in [0, 700\text{ms}) \\ I_{\text{hyp}} + I_{\text{dep}}^{(\text{in})} & t \in [700\text{ms}, 2700\text{ms}) \\ I_{\text{hyp}} & t \in [2700\text{ms}, 3000\text{ms}] \end{cases} ,$$

where the hyperpolarizing current amplitude is $I_{\text{hyp}} = -0.0240728$ nA and the depolarizing current amplitude is $I_{\text{dep}}^{(\text{in})} = 0.3314559$ nA. These values are extracted from the experimental current trace (lower panel in Figure 10.1a) by averaging the input current values in each regime.

In addition, two other experimental traces were taken into account in order to test the data assimilation algorithms forecasts. These are represented in Figure 10.2, and they are the responses to the input currents $I_{\text{dep}}^{(\text{out}1)} = 0.5522364$ nA (the first out-of-sample trace, Figure 10.2a) and $I_{\text{dep}}^{(\text{out}2)} = 0.6999846$ nA (the second out-of-sample trace, Figure 10.2b). The hyperpolarizing step current assume the same value $I_{\text{hyp}} \approx -0.024$ for all traces, although small $\mathcal{O}(10^{-4})$ -deviations are possible because of the averaging of experimental traces. Finally, Figure 10.2c shows the raster plot relative to the in-sample and both the out-of-sample traces, allowing a visual comparison between the respective spike timing. For instance, we note that incrementing the depolarizing current amplitude from $I_{\text{dep}}^{(\text{in})} = 0.3314559$ (in-sample trace) to $I_{\text{dep}}^{(\text{out}2)} = 0.6999846$

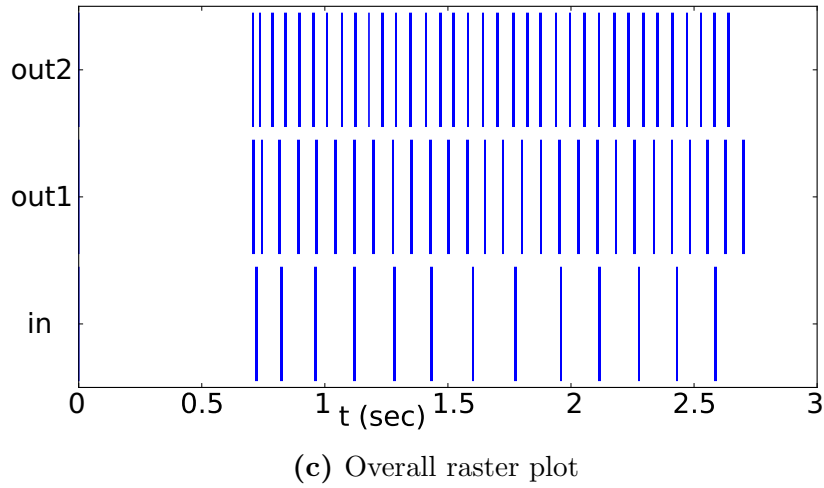
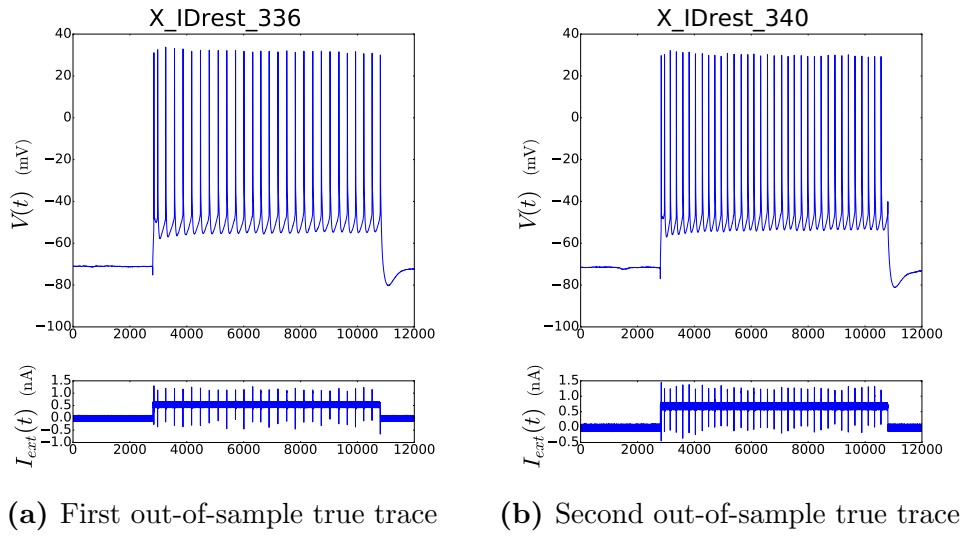


Figure 10.2: Experimental dataset for assimilation: plots of the two out-of-sample experimental traces (a-b) and raster plot of the in-sample trace and both out-of-sample traces (c).

(second out-of-sample trace) results in an increased number of output spikes. Indeed, the spike rate increases from 6.5 Hz (13 spikes in the two-seconds-long injection time for the in-sample experimental trace) up to a 17.5 Hz for the second out-of-sample trace.

10.2 Choosing the parameter search space

First of all, we tried to apply the ensemble Kalman filter to the experimental dataset pictured in Figure 10.2c in similar conditions to the twin experiments described in the previous chapter. That is to say that only positivity constraints were considered on the modelling parameters. As a consequence, the parameter state space was given by the first orthant of the d_θ -dimensional space $\Theta = (0, +\infty)^{d_\theta}$ in this bootstrap attempt. Without going into too many details on the EnKF settings, we just report that we first assumed that the parameter initial condition was centred on the Blue Brain Project value $\mu_{\theta_0} = \theta_{\text{BBP}}$.

As this never yielded any acceptable output (all output traces never exhibited more than one spike at onset of the excitatory input), we also tested different parameter initializations, not only changing the mean but also testing different variance amplitudes. Unfortunately, despite testing an uncountable number of setting parameters configurations, no sufficiently good result was obtained in such conditions. Then, we decided to try overcome such issue by shrinking the search space for the parameter vector Θ in some meaningful fashion.

Therefore, at the first attempt we resorted to the parameter bounds the Blue Brain Project considered in its optimisation procedure. These are not explicitly reported in [149], but they can be recovered from the newly implemented Blue Brain optimiser `bluepyopt` [**BLUE:werner'bluepyopt**], which is available at the web page [9]. We summarise the BBP parameter search intervals in Table 10.3 and henceforth refer to any of such intervals as $I_{\text{BBP}}^{(k)}$, where the integer k is intended to index the components of the parameter vector θ . Reading such table we notice, for instance, that the sodium and potassium conductances have search interval rather large for both somatic and axonal compartments (up to 4000 mS/cm² for axonal \bar{g}_{Nat} and \bar{g}_{Nap}), especially when compared to the calcium conductances.

However, despite the precaution of reducing the search space to a bounded set, the estimation results were still highly unsatisfactory. Such reiterated failure caused the necessity to understand what was going awry with the EnKF, and to possibly identify some way of further reducing the parameter bounds in an effective manner.

10.2.1 Likelihood profiles

Driven by the idea that Bayesian methods should in principle converge to the vicinity of some absolute minimum θ_{opt} of the posterior distribution (which is the case at least for smoothing methods), we decided to look at the likelihood marginal profiles. Indeed, the likelihood term gives a substantial contribu-

Parameter	Somatic $I_{\text{BBP}}^{(k)}$	Axonal $I_{\text{BBP}}^{(k)}$	Apical $I_{\text{BBP}}^{(k)}$	Unit
\bar{g}_{Nat}		[0, 4000]		mS/cm ²
\bar{g}_{Nat2}	[0, 1000]		[0, 40]	mS/cm ²
\bar{g}_{Nap}		[0, 4000]		mS/cm ²
\bar{g}_{Kt}		[0, 100]		mS/cm ²
\bar{g}_{Kp}		[0, 1000]		mS/cm ²
\bar{g}_{M}			[0, 1]	mS/cm ²
$\bar{g}_{\text{Kv3.1}}$	[0, 1000]	[0, 2000]	[0, 40]	mS/cm ²
\bar{g}_{SK}	[0, 100]	[0, 100]		mS/cm ²
\bar{g}_{CaHVA}	[0, 1]	[0, 1]		mS/cm ²
\bar{g}_{CaLVA}	[0, 10]	[0, 10]		mS/cm ²
γ	[0.0005, 0.05]	[0.0005, 0.05]		1
decay	[20, 1000]	[20, 1000]		ms

Table 10.3: Search parameter intervals $I_{\text{BBP}}^{(k)}$ for the optimiser considered in the Blue Brain Project according to the examples implemented in [9]. Only the parameters that were considered free to fit the model in the BBP are reported (for instance, no basal parameter appears in the table because all the modelling parameters of the basal compartment were considered fixed in the BBP). The complete list of model L23_PC_cADpyr229_1 parameters and the corresponding values computed in the Blue Brain Project are given in Table 5.7.

tion to the smoothing posterior (2.12), and it might very well be that some large “energy barrier” separates the initial condition (which is close to θ_{BBP}) from the target minimum θ_{opt} . This statement is motivated in more detail in Remark 10.1.

To this end, we introduce the average sum of squared errors (SSE)

$$d_{\text{SSE}}(\theta, y_{0:J}) = \frac{1}{J+1} \sum_{j=0}^J |y_j - V_j(\theta)|^2. \quad (10.1)$$

In the above expression, we denote $V_{0:J}(\theta)$ the deterministic solution of model L23_PC_cADpyr229_1 (equations (5.11), (5.12), (5.13), and (5.14)) with modelling parameters given by θ and a deterministic initial condition given by $V_0 = y_0 \approx -70\text{mV}$. The initial values for activation and inactivation variables are given by the corresponding asymptotic values $m_{\text{ion},\infty}(V_0)$ and $h_{\text{ion},\infty}(V_0)$ and the value of 5×10^{-5} mM is taken as initial calcium concentration¹. With

¹Note that in the following Remark 10.1, the shorthand $x_0(V_0)$ is used to denote the complete initial state variable vector defined as a function of the initial membrane potential V_0 according to such asymptotic-value procedure.

such definition of $V_{0:J}(\theta)$, we can look at the previous expression as a sort of distance between the parameter vector θ and the experimental trace $y_{0:J}$: the smaller $d_{\text{SSE}}(\theta, y_{0:J})$ gets, the closer θ is to the target parameter set θ_{opt} .

Remark 10.1. *Note that the function $\theta \mapsto d_{\text{SSE}}(\theta; y_{0:J})$ defined in (10.1) is proportional to the (neg-log-) smoothing distribution (2.12) of the augmented state space $(x, \theta) \in \mathcal{X} \times \Theta$ for model `L23_PC_cADpyr229_1`, in case of deterministic dynamics.*

In fact, suppose that $\Sigma_z = \text{diag}(\Sigma_x, \Sigma_\theta) = 0$ (deterministic dynamics in both x and θ), that $C_{z_0} = \text{diag}(C_{x_0}, C_{\theta_0}) = 0$ (the initial condition is deterministic), and that the measurement covariance matrix is given by $\Gamma^{1/2} = 1$ mV. Then, since the neg-log-smoothing function (2.12) writes²

$$\begin{aligned} \mathcal{S}(x_{0:J}, \theta_{0:J}; y_{0:J}) &= \frac{1}{2} \delta_{x_0(y_0)}(x_0) + \frac{1}{2} \sum_{j=1}^J \delta_{g_{j-1}(x_{j-1})}(x_j) + \frac{1}{2} \sum_{j=0}^J \delta_{\theta_0}(\theta_j) \\ &\quad + \frac{1}{2} \sum_{j=0}^J |y_j - V_j(\theta_0)|^2, \end{aligned}$$

we have that the posterior function only consists in the likelihood term. Indeed, all Dirac delta terms can be seen as simple constraints. However, we notice that in the r.h.s. of the above equation, the last sum can be written as the squared L^2 -distance $d_2(V_{0:J}(\theta_0), y_{0:J})$. Hence, in case of Gaussian state-space model the likelihood derives from the Euclidean distance on the measurement space $\mathbb{R}^{d_y(J+1)}$ which, in return, is proportional to the average SSE.

In conclusion, we just showed that minimizing $d_{\text{SSE}}(\cdot, y_{0:J})$ is equivalent to find the maximum of the smoothing distribution in case of deterministic dynamics and Gaussian state-space model. ♠

For $\theta^k = \bar{g}_{\text{Nat}2}, \bar{g}_{\text{Kv}3.1}, \bar{g}_{\text{SK}}, \bar{g}_{\text{CaHVA}}, \bar{g}_{\text{CaLVA}}, \bar{g}_{\text{L}}, \bar{g}_{\text{H}}, \gamma, \text{decay}$, the generic k -th panel of Figure 10.4 plots the marginal likelihood profile in case all model parameters but the k -th are fixed to the value compute by the BBP, while the k -th parameter spans the Blue Brain Project search interval $I_{\text{BBP}}^{(k)}$ in a univariate manner. Note that all BBP search intervals are listed in Table 10.3, except for those of parameters \bar{g}_{L} and \bar{g}_{H} which were considered fixed in the BBP. In our experiments, we chose these two search spaces to be both equal to $[0 \text{ mS/cm}^2, 0.01 \text{ mS/cm}^2]$.

In formulas, the k -th panel pictures the graph of the marginal **univariate d_{SSE} -fit function**

$$\theta^k \mapsto d_{\text{SSE}}(\theta_{\text{BBP}}^{(-k)}, y_{0:J})$$

²We recall that we use the notation $\delta_{\tilde{x}}(x)$ to indicate the Dirac mass centred on the singleton $\{x = \tilde{x}\}$.

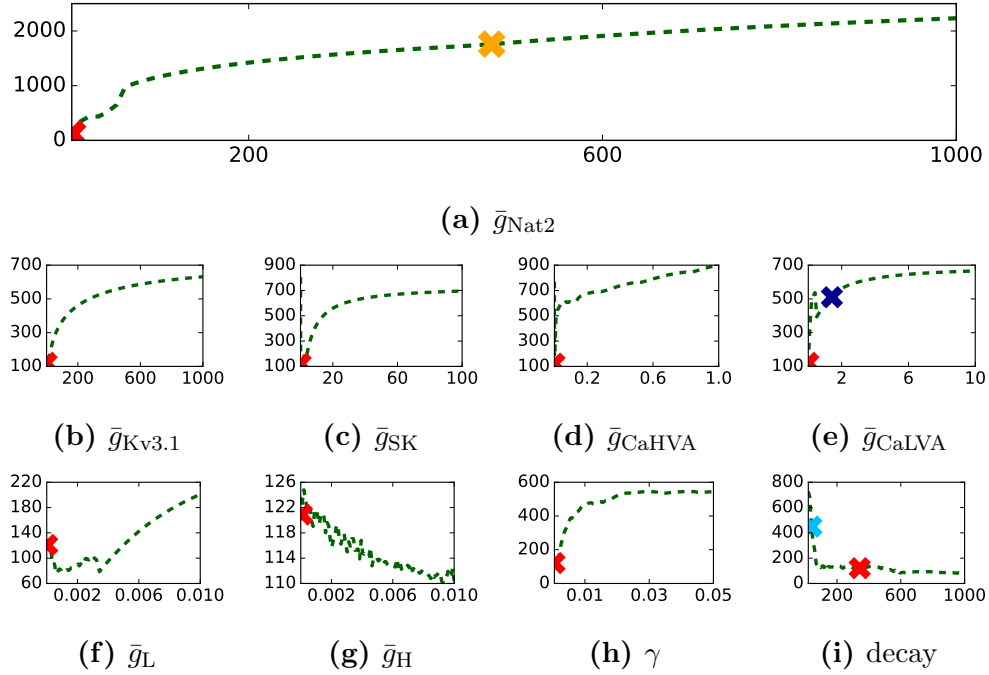


Figure 10.4: Graphs of the univariate d_{SSE} -fit functions, i.e. the marginal average sums of squared errors $I_{\text{BBP}}^{(k)} \ni \theta^k \mapsto d_{\text{SSE}}(\theta_{\text{BBP}}^{(-k)}, y_{0:J})$. The red crosses mark the position of the parameter value computed from the BBP, while the orange, blue and light blue crosses point out the parameter values that produced the traces in Figure 10.5.

for $\theta^k \in I_{\text{BBP}}^{(k)}$, where $\theta_{\text{BBP}}^{(-k)} = (\theta_{\text{BBP}}^1, \dots, \theta_{\text{BBP}}^{k-1}, \theta^k, \theta_{\text{BBP}}^{k+1}, \dots, \theta_{\text{BBP}}^{d_\theta})$. A red cross is reported in each panel to point out the parameter value θ_{BBP}^k .

The first straightforward observation is that the red crosses are all located at the far left of the search intervals, corresponding to conductances which are very small with respect to the overall size of the search space. In addition, it appears that the likelihood is small only in a neighbourhood of the BBP value, whereas it gets almost constant approaching the supremum of the search space.

To investigate the reasons of such parameter-space structure, we plotted the traces corresponding to some of the parameter large values. The specific values are those corresponding to the orange, the dark blue, and the light blue crosses in Figure 10.4a, Figure 10.4e, and Figure 10.4i, respectively. The resulting trajectories are plotted in Figure 10.5, and we immediately note that most of them do not really represent any plausible neuronal activity. In particular, the dark blue trace and the orange traces (which result from setting all model parameter to the BBP values, except for \bar{g}_{CaLVA} and \bar{g}_{Nat2} , respectively) only produce one action potential and then exhibit a somewhat unrealistic resting

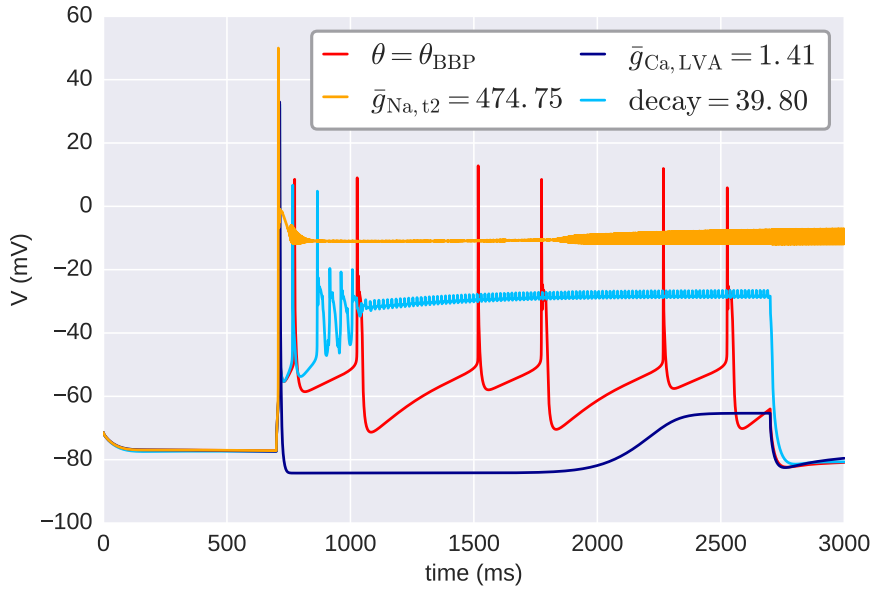


Figure 10.5: Numerical solutions of model L23_PC.cADpyr229_1 for different values of the modelling parameter θ . The red trace exhibiting seven spikes corresponds to $\theta = \theta_{\text{BBP}}$, whereas the other lines are produced modifying one single component of the parameter vector (see the legend).

behaviour.

Then, we also plot the graph of the **univariate D_S -fit function**

$$\theta^k \mapsto D_S(\theta_{\text{BBP}}^{(-k)}, y_{0:J}),$$

where D_S is the SPIKE-distance defined in (7.2) and the distance between the parameter $\theta_{\text{BBP}}^{(-k)}$ and the dataset $y_{0:J}$ is intended in the same sense as in (10.1). Analysing Figure 10.6, we remark that the plateau effect is even more pronounced. Since that SPIKE-distance captures more accurately the electrophysiological activity than SSE, we deduce that the electrophysiology properties are substantially constant in that region. Hence, it looks like in a very large portion of the search space, the model simply does not represent an active neuron when the in-sample input $I_{\text{ext}}^{(\text{in})}(t)$ is presented!

We draw the conclusion that a large portion of the parameter space is actually irrelevant to our search. Indeed, in such parameter region the model seems to be no plausible candidate to reproduce the neuron activity we feed it. In the following section we describe our choice of a criterion for reducing the parameter search space size.

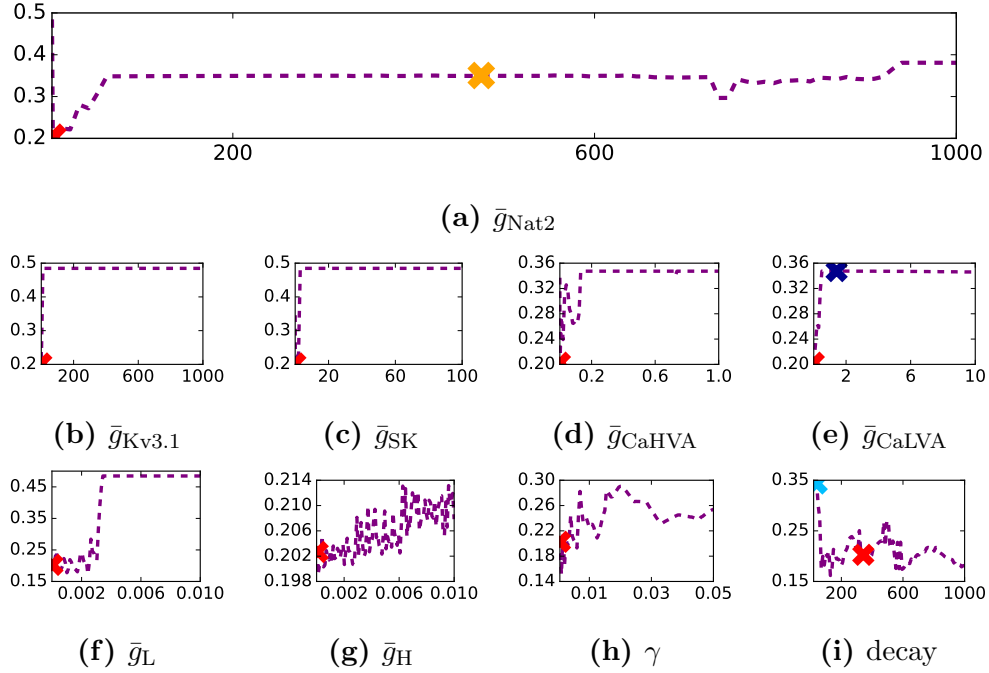


Figure 10.6: Graphs of the univariate D_S -fit functions, i.e. the marginal SPIKE-distances $I_{\text{BBP}}^k \ni \theta^{(k)} \mapsto D_S(\theta_{\text{BBP}}^{(-k)}, y_{0:J})$. The red crosses mark the position of the parameter value computed from the BBP, while the orange, blue and light blue crosses point out the parameter values that produced the traces in Figure 10.5.

10.2.2 Reducing the parameter search space size

Analysing Figure 10.5, we choose to base our search-space shrinkage on whether or not model L23_PC_cADpyr229_1 produces realistic responses in a given parameter region. Specifically, in order to perform such analysis in a non-subjective manner, we decide to tell apart a plausible parameter configuration from a non-realistic one based on whether the number of spikes elicited by the input current $I_{\text{ext}}^{(\text{in})}(t)$ is smaller or larger than three.

Indeed, we report that most of the traces exhibiting less than three action potentials that we inspected were similar to the orange and the dark blue traces in Figure 10.5. These exhibit only one spike³ and then display a strange resting behaviour which we consider as too odd to represent a neuron plausible response to an excitatory input. We think that searching the parameter values in those portion of the parameter intervals would be a waste of time. On the contrary, we deem desirable to leave enough room in the search space

³Using the `elephant` Python module, an action potential is detected only when the membrane potential surpasses the 0 mV value and changes sign.

to include traces like the light blue one. Indeed, although this one exhibits a peculiar behaviour, it is still close to a plausible neural response to $I_{\text{ext}}^{(\text{in})}(t)$. In fact, although the potential trace is significantly ragged, it could be that modifying other parameters the choice $\text{decay} = 39.80$ ms would result in realistic trajectories.

Since, as hinted by Figure 10.4 and Figure 10.6, we observed a monotony in the number of spikes, we argue that identifying the parameter values θ^k such that $V_{0:J}(\theta_{\text{BBP}}^{(-k)})$ exhibits exactly three action potentials is enough to effectively reduce the search-space size. In order to do so in an automated manner, we take advantage of the Victor-Purpura SPIKE distance introduced in Section 7.1.1 using $q = 0$. As for such value of the metric parameter only the number of spikes are compared, in this chapter we write D^{COUNT} instead of $D_{q=0}^{\text{SPIKE}}$ for the sake of notation compactness. In particular, we compare the spike train resulting from a given trace $V_{0:J}(\theta)$ to an empty spike train (denoted 0 here), and look for the zeros of the univariate function

$$\theta^k \mapsto D^{\text{COUNT}}(\theta_{\text{BBP}}^{(-k)}, 0) - 3.$$

Again, this means that we look for the parameter value such that the resulting model exhibits exactly three action potential. We stress that this is the only condition we impose to discriminate between realistic and unrealistic parameter values: if for a given parameter set the model produces less than three spikes, then the model's response to $I_{\text{ext}}^{(\text{in})}(t)$ is considered as unrealistic and that parameter set is hence discarded.

In practice, we choose to apply a bisection method with absolute tolerance $\text{tol} = 10^{-3}$ (i.e., the θ^k zero-point value has to be accurate to the third decimal digit). We chose the bisection method as we are dealing with a discrete-valued function (D^{COUNT} only assumes integer values) so no derivative-based method or secant-like method would be particularly effective. Note that, theoretically speaking, the bisection method only applies to continuous functions. However, we assume that the Victor-Purpura COUNT distance is continuous in an integer-sense (i.e., for small changes in a single parameter value θ^k , the function varies smoothly assuming all intermediate integer values), so that the bisection method is guaranteed to converge to some zero of the target function.

In fact, $\theta^k \mapsto D^{\text{COUNT}}(\theta_{\text{BBP}}^{(-k)}, 0) - 3$ assumes opposite sign at the boundaries of $I_{\text{BBP}}^{(k)}$ for most parameters. In particular, for all parameters but $\bar{g}_{\text{Nat}2}$, decay , and \bar{g}_{H} the function is positive in the lower boundary and negative (i.e., it produces less than three spikes) in the upper boundary. On the other hand, decay and \bar{g}_{H} always produce realistic traces (the respective univariate D^{COUNT} functions assume positive values at both edges), whereas the univariate D^{COUNT} function relative to $\bar{g}_{\text{Nat}2}$ is negative at both boundaries. For the latter, we pro-

Parameter	Somatic $I_{\text{new}}^{(k)}$	Somatic $I_{\text{BBP}}^{(k)}$	Unit
\bar{g}_{Nat2}	[0.463353, 6.78065]	[0, 1000]	mS/cm ²
$\bar{g}_{\text{Kv3.1}}$	[0, 1.95312]	[0, 1000]	mS/cm ²
\bar{g}_{SK}	[0, 0.585938]	[0, 100]	mS/cm ²
\bar{g}_{CaHVA}	[0, 0.00585938]	[0, 1]	mS/cm ²
\bar{g}_{CaLVA}	[0, 0.136719]	[0, 10]	mS/cm ²
γ	[0.0005, 0.00755762]	[0.0005, 0.05]	1
decay	[20, 1000]	[20, 1000]	ms
\bar{g}_{L}	[0, 0.00112305]	[0, 0.01]*	mS/cm ²
\bar{g}_{H}	[0, 0.01]	[0, 0.01]*	mS/cm ²

Table 10.7: Updated search parameter intervals $I_{\text{new}}^{(k)}$ compared to those of the Bue Brain Project $I_{\text{BBP}}^{(k)}$. Only somatic parameters considered in our data-assimilation experiment are reported. The intervals marked with an asterisk (*) were not considered by the BBP but arbitrarily set by the authors.

ceeded running the bisection method on two separate intervals: the first interval ranges from the lower bound of \bar{g}_{Nat2} 's $I_{\text{BBP}}^{(k)}$ and $\theta_{\text{BBP}}^{(k)} = 0.926705$ mS/cm², while the second ranges from $\theta_{\text{BBP}}^{(k)}$ to the upper boundary of \bar{g}_{Nat2} 's $I_{\text{BBP}}^{(k)}$. This is possible, because we know from Figure 10.5 that for $\theta = \theta_{\text{BBP}}$ such function is greater than zero (the red trace exhibits seven action potentials). Such bisection procedure resulted in the updated search intervals for the somatic parameters $I_{\text{new}}^{(k)}$ which are presented in Table 10.7.

Note that we could have included axonal and apical parameters as well. However, in all tests we ran the influence of non-somatic parameters on the univariate D_S - and the d_{SSE} -fit functions was several order of magnitude smaller than the somatic ones. Then, we concluded that assessing non-somatic parameters would not be meaningful without including non-somatic data in the assimilation.

Remark 10.2. *We acknowledge that the bisection approach we propose here is far from being ideal. Some of the main flaws we identified include the following.*

- *The main choices we made to come up with such procedure are extremely application-dependent. In particular:*
 - ▶ *only responses to the input current $I_{\text{ext}}^{(\text{in})}(t)$, starting from the single specific initial condition $x_0(y_0)$ are tested;*
 - ▶ *assessing whether or not a model is able to represent a neuron only based on the number of action potentials it produces is a choice which neglects several aspects of a neuron electrophysiology.*

- *The algorithm output depends on the “base point” θ_{BBP} . Using a different parameter set as base point to univariately modify one parameter at the time would result in different updated intervals.*
- *It is possible that there are multiple zeros and we only took the first one reachable via the bisection method.*
- *Our procedure only considers a univariate approach. However, note that a more complete multivariate analysis would be too expensive from a computational point of view. Indeed, sampling the whole multidimensional parameter space with a sufficiently dense mesh requires a huge number of function evaluations. For instance 100 sample points per univariate interval would result in $\mathcal{O}(10^{18})$ function evaluations. To give an idea, considering that one hundred function evaluations (i.e. one hundred model runs) take about $3 \text{ min } 3\text{s} \pm 43\text{s}$ in average⁴, this would result in an execution time of about ten billion years, approximately twice the age of planet Earth. This is a clear example of the curse of dimensionality.*

Nevertheless, such approach has the advantage of being simple, fast to be executed, and, most importantly, it allows one to identify a good set of parameters fitting the experimental trace in some sense, as we show in the following sections. ♠

After this adjustment, we ran the EnKF enforcing the new reduced bounds $I_{\text{new}}^{(k)}$'s. However, all output traces turned out being absolutely unrealistic once again, so that it is not even worth reporting them here. Then, since at this point we got convinced the sequential algorithm was not effective to assimilate experimental data in model L23_PC_cADpyr229_1, we decided to double back to some kind of variational method to explore the parameter search space in a more direct way. In particular, we decided to apply a brute-force minimization method of some cost function of choice. In the following section we describe and motivate our final decision for such an objective function.

10.3 Selecting an effective assimilation method

In practice, we tested a number of objective functions and, in analogy to the procedure which motivated the search-space size reduction, the first natural

⁴Value estimated running one hundred parameter configurations for 2500 ms in parallel on the Fudan University Institute for Science and Technology for Brain-Inspired Intelligence (ISTBI) computing cluster ten times. The cluster specifics are those reported at the beginning of Section 9.1

choice was to select the multivariate SSE-distance $\theta \mapsto d_{\text{SSE}}(\theta, y_{0:J})$, or some $\ell^p(\mathbb{R}^{d_\theta})$ -distance (i.e., a sort of parameter-equivalent of the absolute measurement error defined in Section 4.4.1). Unfortunately, our preliminary studies seemed to highlight that such metrics were not effective at all, even though we tested different values of the distance parameters p .

In a second attempt, we also tried to augment the solution space with the time derivatives (approximated via finite-difference methods) of both $V_{0:J}(\theta)$ and of $y_{0:J}$, so to equip the ℓ^p -comparison of the two traces with some information regarding the time placement of action potentials. In fact, spikes consists in a sudden depolarization followed by the swift hyperpolarization of the neuron membrane potential, and such increase and decrease pattern has a sharp counterpart into both the magnitude and the sign of the potential difference quotient $\frac{V_{j+1}-V_{j-1}}{2\Delta t}$. Disappointingly, not even this expedient improved the performances with respect to the previously tested ℓ^p distance.

Then, we redirected ourselves to some metric more tailored to detect differences in two single-neuron activities. In particular, we dropped the ambition to point-wisely estimate the membrane potential trace and focused on the following spike train distances:

- the van Rossum distance, defined in Section 7.2.1, resulting in the **van Rossum objective function**

$$\theta \mapsto D_q^{\text{ROSS}}(\theta, y_{0:J});$$

- the Victor-Purpura D_q^{SPIKE} SPIKE train distance defined in Section 7.1.1, resulting in the objective function

$$\theta \mapsto D_q^{\text{SPIKE}}(\theta, y_{0:J}),$$

which we refer to as the **Victor-Purpura objective function**;

- and the parameter-free SPIKE distance defined in Section 7.3.1, corresponding to the **SPIKE-distance objective function**

$$\theta \mapsto D_S(\theta, y_{0:J}).$$

In all results we present in the following, we took advantage of the size reduction presented in the previous section. Namely, the parameter search-space was set to be the d_θ -dimensional hyper-rectangle $\Theta = \prod_{k=1}^{d_\theta} I_{\text{new}}^k$.

Remark 10.3. *Note that, using the same notation established at the beginning of this chapter, when we mention the spike-train cost functions it is implied*

that the distance is computed between the respective spike-trains. For instance, $D_q^{\text{SPIKE}}(\theta, y_{0:J})$ denotes the Victor-Purpura spike distance between the spike train extracted from the membrane potential trace $V_{0:J}(\theta)$ and the spike train extracted from the experimental trace $y_{0:J}$. ♠

One may think that minimizing such objective functions essentially consists in resorting to the 4DVAR smoothing method. However, this is actually true only if we assume a uninformative prior distribution on the signal augmented variable.

However, even assuming we are applying 4DVAR, resorting to minAone code was not an option for implementation reasons. In fact, the code developed by H. Abarbanel's research team requires that the complete ODE vector field underlying the deterministic discrete-time map $g_{j-1}(x_{j-1}, \theta_{j-1})$ can be explicitly written in symbolic form. Although this is possible in many applications, it is not the case for the Neuron-implemented BBP models. In fact, Neuron uses the compiled programming language nmodl⁵ to build its neuron models, and linking the Python scripts which include the resulting executables to the IPOPT minimization toolbox minAone hinges upon is not straightforward. As a consequence, we decided to take advantage of some other Python optimisation toolbox.

However, since at this point we wished to use some ready-to-use library, we first tested simulated annealing [204], a global metaheuristic and probabilistic optimisation method which is related to Metropolis-Hastings methodology to a certain extent. However, in addition to requiring a non-negligible fine tuning of its setting parameters, such method appeared to converge extremely slowly and it did not provide any useful solution in all runs we launched.

Then, we tested a different family of optimisation methods, namely metaheuristic population-based algorithms. In particular, we took advantage of the efficient implementation of the particle swarm optimisation method (PSO, see Remark 2.8) provided by the parallel Python module `pygmo` [25], developed by F. Biscani and D. Izzo.

In the following section, we analyse the results obtained minimizing the above-mentioned objective functions with the PSO methods .

10.4 Results analysis

In particular, we applied the canonical particle swarm method with constriction factor, corresponding to variant 5 of algorithm `PyGMO.algorithm.pso`

⁵the Neuron version of the model description language (modl) [87], developed by the NBSR (National Biomedical Simulation Resource, [119])

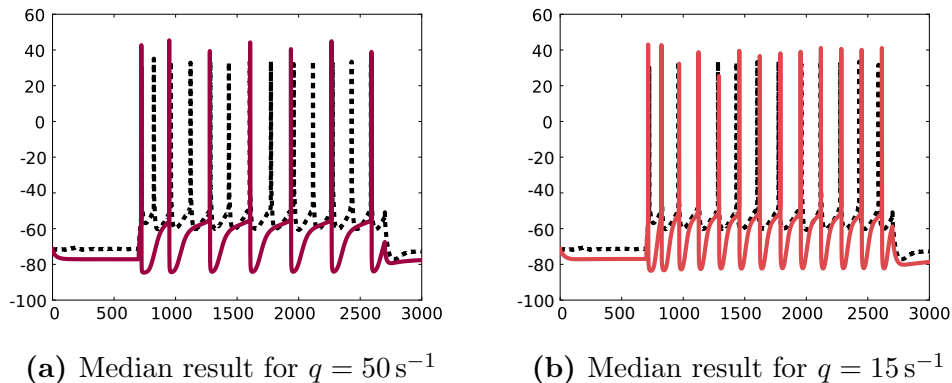
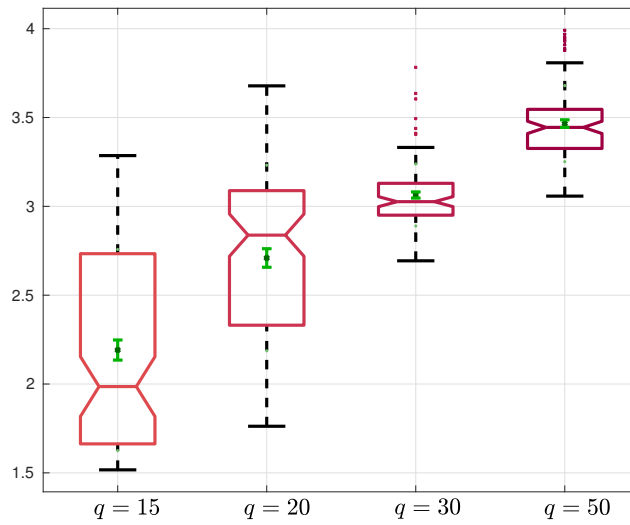


Figure 10.8: Median result of van Rossum objective function for (a) $q = 50 \text{ s}^{-1}$ and for (b) $q = 15 \text{ s}^{-1}$

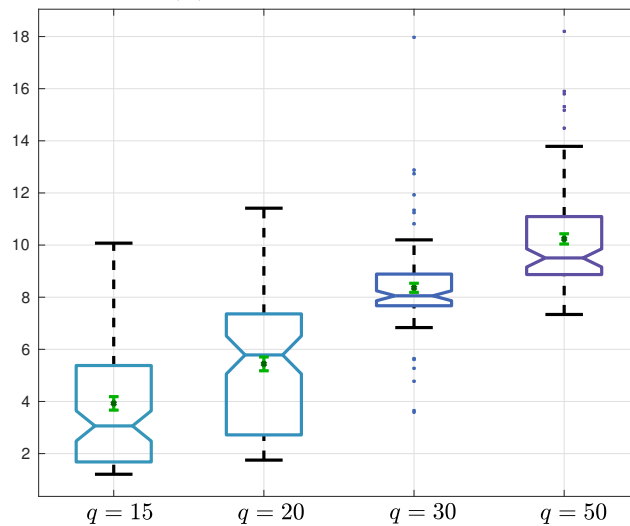
(refer to [168] for more details on such method) using the following values of the method parameters: constriction factor $\omega = 0.7298$, cognitive component $\eta_1 = 2.05$, social component $\eta_2 = 2.05$, and maximum velocity coefficient $v_{\text{coeff}} = 0.5$ (refer to [10] for more details on the `pygmo` implementation of the PSO; such website also provides a general overview of the `pygmo` Python module, its essential documentation, and some use examples). For all objectives, we considered 100 independent runs of the PSO, using a swarm size of 25 particles, evolving for up to 150 generations.

First, we tested if the van Rossum objective function provided sufficiently good solutions. Since in principle the larger the q -value, the more sensitive the metric is to precise spike timing, we started off by selecting $q = 50 \text{ s}^{-1}$. A trace representative of the resulting general output is represented Figure 10.8a, where the estimated membrane potential time course corresponding to the median $D_{q=50}^{\text{Ross}}$ -value is plotted against the assimilated experimental data. It is evident that the average result produced by such metric is not satisfactory at all, as the timing of about one action potential out of two is correctly estimated. Nevertheless, we remark that the results obtained with such objective function are still far better than those produced by any of the unreported attempted methods (from EnFK, to previously tested minimization methods or metrics).

Then, we tested different values of the scale parameter q , to check if this was enough to improve the estimation quality. The overall results for $q = 15 \text{ s}^{-1}$, 20 s^{-1} , 30 s^{-1} , 50 s^{-1} are presented in compact form through the box-plots pictured in Figure 10.9a. These highlights that selecting $q = 15 \text{ s}^{-1}$ actually allows the minimization method to reach considerably smaller objective values than any other tested q -value. The statistical significance of this statement was confirmed by the application of a Tukey's HSD test with $p < 5\%$ (note, in this



(a) van Rossum distance



(b) Victor-Purpura distance

Figure 10.9: Box-plot comparison of the in-sample results obtained minimizing (a) the van Rossum, and (b) the Victor-Purpura objective functions with $q = 15 \text{ s}^{-1}, 20 \text{ s}^{-1}, 30 \text{ s}^{-1}, 50 \text{ s}^{-1}$, ranked according to the respective fitness values. The box lower and upper edges mark the first and third quartile, respectively, while the bar in the middle of the box denotes the median objective function value and the notches its confidence interval. The whiskers extend up to 1.5 times the interquartile range, and the points falling outside such ranges are considered outliers of the distribution. The dark green crosses denote the distribution mean, while the light green error bars their confidence intervals.

sense, the non-intersecting confidence intervals of the group means marked by light-green error-bars). In term of membrane potential traces, this translates in the substantial improvement of the matching between the estimated spike-train and the experimental one pictured in Figure 10.8b.

Aiming at identifying the objective function able to obtain the best estimation results, we first performed the same test on the Victor-Purpura objective function. Again, we obtained that $q = 15 \text{ s}^{-1}$ is the best choice for such metric too (see Figure 10.9b). Then, we took the SPIKE-distance objective function into consideration. In order to compare the results obtained with the best q -value for both the van Rossum and the Victor-Purpura objective function to the SPIKE-distance ones, we needed a single comparison criterion. Being only parameter-free and uniformly bounded metric, we opted for ranking all previously found solutions according to the D_S -metric. This simply means that we computed the performance scores $D_S(\text{in})$, $D_S(\text{out1})$, and $D_S(\text{out2})$ for all traces produced minimizing each of the cost functions (D_S , $D_{q=15}^{\text{Ross}}$, and $D_{q=15}^{\text{SPIKE}}$), and used such values to compare the quality of each run. Note that, since, as we show in the following, once the suitable scale parameter q has been identified the three metrics yield very similar results, we argue that using any of these metrics to rank and compare the results would actually lead to analogous outcomes. Nevertheless, we claim that the normalization considered in the $[0, 1]$ -valued D_S distance allows one to better compare the response to stimuli with different amplitudes. Indeed, both van Rossum and Victor-Purpura SPIKE distances are approximately linear in the number of spikes contained in the spike train, and the comparing the response in, say, the **out1** and **out2** time window would lead to a larger bias. In fact, the different time windows we considered contain a significantly different number of spikes. Note that, before proceeding in the comparison, we first verified that $q = 15 \text{ s}^{-1}$ is the best value for both van Rossum and Victor-Purpura results also when measuring the fitness-value according to the SPIKE-distance.

First, running a non-parametric Friedman test on the in-sample fitness values $D_S(\text{in})$ we observed that no statistically significant difference holds between the results produced by these three metrics. Then, considering the average error among all time windows $[D_S(\text{in}) + D_S(\text{out1}) + D_S(\text{out2})]/3$, we found that the distribution of the predicted results across all time windows is essentially the same (see Figure 10.10). All these evidences highlight that, for the three cost functions considered, the procedure fits the assimilated data reaching fundamentally the same in-sample precision and similar out-of-sample predictions.

Furthermore, we found that unlike the biased twin experiment results, a small in-sample error is not a good predictor for a good out-of-sample esti-

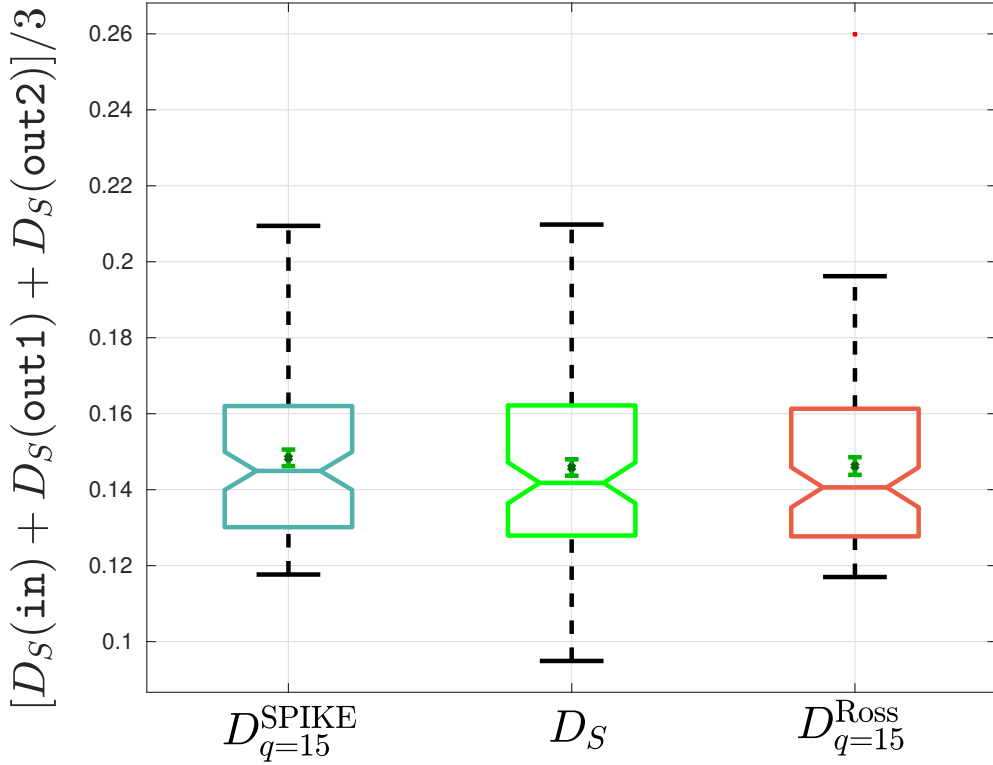


Figure 10.10: Empirical p.d.f.'s of the $D_{q=15}^{\text{SPIKE}}$ results (red line), the D_S results (bright green line), and the $D_{q=15}^{\text{Ross}}$ results (blue line). The x -axis represents $[D_S(\text{in}) + D_S(\text{out1}) + D_S(\text{out2})]/3$, i.e. the average value of the SPIKE-distance error in the in-sample and both out-of-sample time windows. Box-plot description as in Figure 10.9.

mation. Indeed, the absolute correlation coefficient $|\rho|$ between the in-sample error $D_S(\text{in})$ and the average out-of-sample error $[D_S(\text{out1}) + D_S(\text{out2})]/2$ is always smaller than 0.05. Then, if using the EnKF a good in-sample score already gives some insight on the quality of the out-of-sample predictions (see, the twin experiment described in Section 9.2 and, in particular, Figure 9.10), using a cost-function minimization approach the successful runs can only be selected based on some out-of-sample prediction.

10.4.1 Selection by validation in a forecast-skill sense

Then, to select the good runs obtained minimizing each of the three spike-train-based cost function, we proceeded by a validation argument as in Section 9.2.4. Namely, we checked how the model endowed of the estimated set of parameters

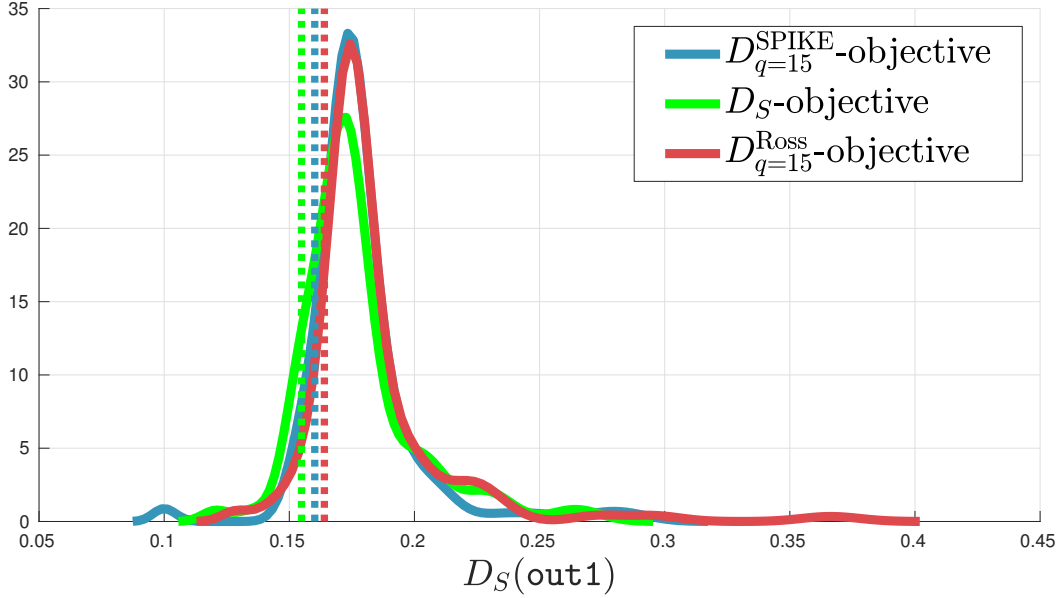


Figure 10.11: Empirical p.d.f.'s of the $D_{q=15}^{\text{SPIKE}}$ results (blue line), the D_S results (bright green line), and the $D_{q=15}^{\text{Ross}}$ results (red line). The x -axis represents $D_S(\text{out1})$, i.e. the SPIKE-distance score in the validation set (the first out-of-sample time window).

$\hat{\theta}$ predicts experimental traces in the first out-of-sample time window `out1` (consisting in experimental data which were not used to train the model).

The empirical p.d.f.'s represented in Figure 10.11 highlight that the resulting $D_S(\text{out1})$ statistics is substantially overlapping for all three cost functions and the profile is extremely similar too. Then, in order to identify the **good runs** produced by each cost function, we considered the best 10 runs in a forecast-skill sense (those with minimal $D_S(\text{out1})$ performance score) for each cost-function. In particular, such runs correspond to the empirical 10-percentile of the $D_S(\text{out1})$ value (0.160 for the $D_{q=15}^{\text{SPIKE}}$ cost function, 0.155 for the D_S cost function, and 0.164 for the $D_{q=15}^{\text{Ross}}$ results, respectively; see the vertical dotted lines in Figure 10.11). Limiting ourselves to these good runs, we were able to compute the prediction error in the second out-of-sample window `out2`. Note that such quantity represents an estimate of the prediction quality of the neuron response to new stimuli. We obtained that the performance score in such test dataset is analogous for all three metrics: for the Victor-Purpura $D_{q=15}^{\text{SPIKE}}$ good runs the $D_S(\text{out2})$ score is 0.171 ± 0.007 , for the SPIKE-dist D_S results it is 0.166 ± 0.024 , and for the van-Rossum $D_{q=15}^{\text{Ross}}$ results 0.174 ± 0.009 . In addition, note that such values are fairly close to the respective 10-percentiles, suggesting that if a parameter set $\hat{\theta}$ predicts an

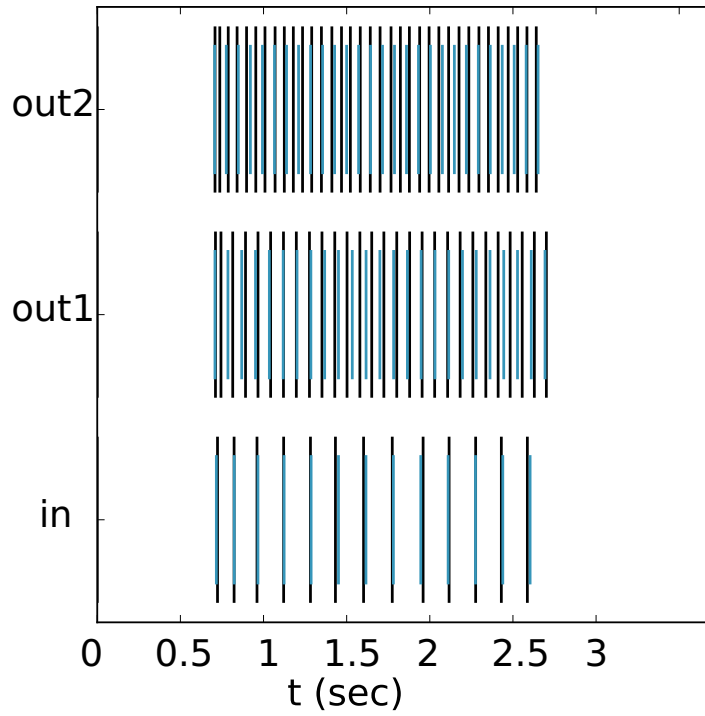
Parameter	Victor-Purpura		SPIKE-distance		van-Rossum	
	mean	cv	mean	cv	mean	cv
\bar{g}_{Nat2}	5.51	0.23	5.79	0.21	4.37	0.38
$\bar{g}_{\text{Kv3.1}}$	0.31	0.67	0.54	0.68	0.38	0.60
\bar{g}_{SK}	0.29	0.59	0.25	0.72	0.30	0.67
\bar{g}_{CaHVA}	0.0035	0.34	0.0039	0.45	0.0037	0.56
\bar{g}_{CaLVA}	0.0479	0.93	0.0278	0.78	0.0306	1.27
γ	0.0029	0.82	0.0034	0.56	0.0033	0.71
decay	20.33	0.03	24.36	0.35	28.95	0.47
\bar{g}_{L}	0.0003	1.02	0.0002	0.65	0.0003	1.37
\bar{g}_{H}	0.0033	1.01	0.0057	0.62	0.0047	0.69

Table 10.15: Mean and coefficient of variation of the parameter values underlying the “good runs” resulting by the minimization of cost functions $D_{q=15}^{\text{SPIKE}}$, D_S , and $D_{q=15}^{\text{Ross}}$.

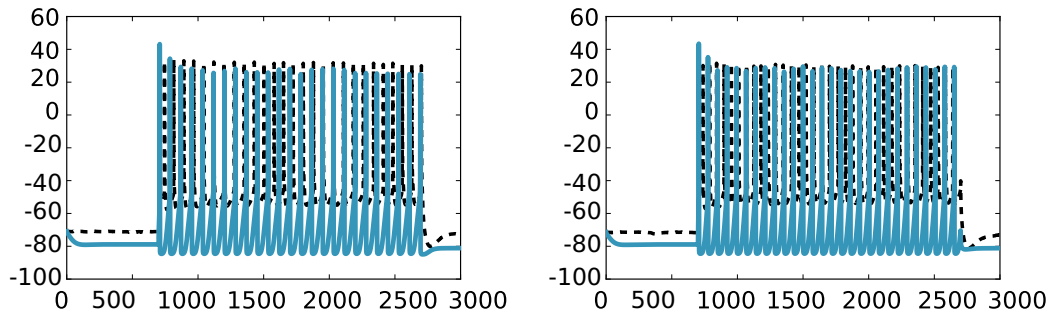
out-of-sample trace accurately, it is reasonable to expect it predicts well other out-of-sample traces too.

To give a graphical representation of the corresponding solutions, we show the raster plot and the underlying out-of-sample predictions of one of the ten selected runs for the Victor-Purpura cost function (Figure 10.12), for the SPIKE-distance cost function (Figure 10.13), and for the van Rossum distance (Figure 10.14). As one can notice, only the spike timing are predicted with good (but not perfect) accuracy whereas the overall time course of the membrane potential is not reproduced by any of the traces. In particular, the base potential value of the predicted traces is lower than the one of the experimental recordings, as well as the value of the after-hyper-polarization depth and the action-potential height.

To investigate the values of the model parameters that underlie the selected runs, in Table 10.15 we report the mean and the coefficient of variation of the estimated parameters in the “successful runs” obtained minimizing each cost function. Analysing the table, we observe that the smallest coefficient of variation are those for parameters decay, \bar{g}_{Nat2} , \bar{g}_{CaHVA} (all smaller or close to 0.5 for all cost functions), and then for $\bar{g}_{\text{Kv3.1}}$, \bar{g}_{SK} and γ (usually between 0.5 and 0.75). On the contrary, the coefficient of variation of \bar{g}_{CaHVA} , \bar{g}_{L} and \bar{g}_{H} appear to be close or larger than one, suggesting that their estimate cannot be considered precise. As a consequence, we argue that the estimates for \bar{g}_{Nat2} , $\bar{g}_{\text{Kv3.1}}$, \bar{g}_{SK} , \bar{g}_{CaHVA} , decay and γ may be considered sufficiently accurate in a forecast-skill sense, meaning that the range of the estimated values is likely to that one which allow the behaviour of the experimental recordings. Indeed,

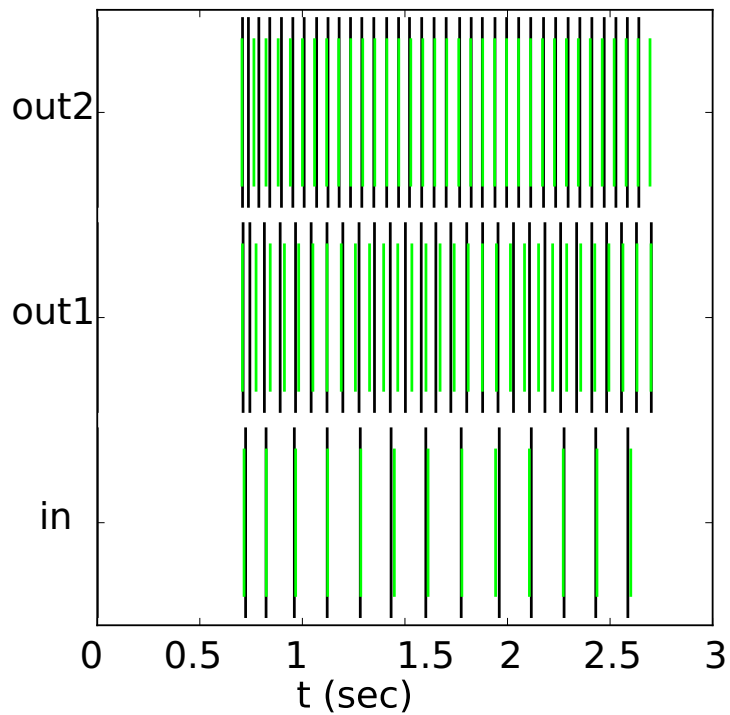


(a) Overall raster plot. The black bars indicate the spike times of the experimental trace $y_{0,J}$, while the blue traces denote the spike times of the estimated trajectory.

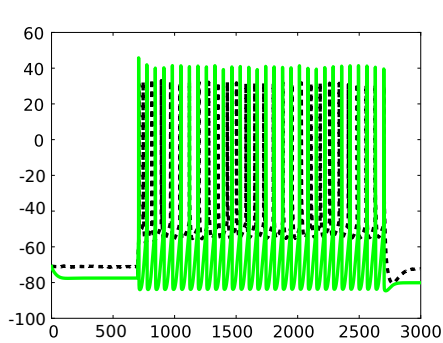


(b) First out-of-sample time window (c) Second out-of-sample time window.

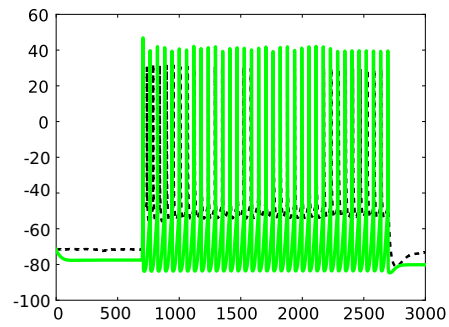
Figure 10.12: Output obtained minimizing the $D_{q=15}^{\text{SPIKE}}$ -objective function representative of the corresponding “good runs” (the lower 10-percentile). In (b) and (c) the blue traces indicate the estimated potential trace, which is plotted against the respective experimental trace (dashed black line).



(a) Overall raster plot. The black bars indicate the spike times of the experimental trace $y_{0:J}$, while the orange traces denote the spike times of the estimated trajectory.

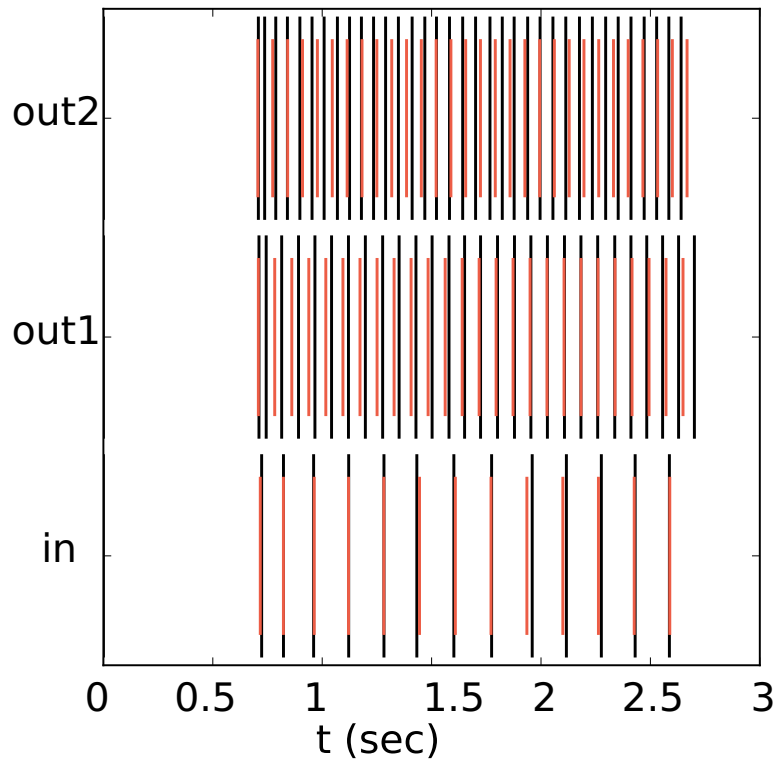


(b) First out-of-sample time window

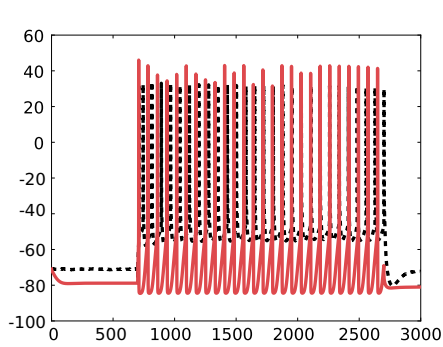


(c) Second out-of-sample time window.

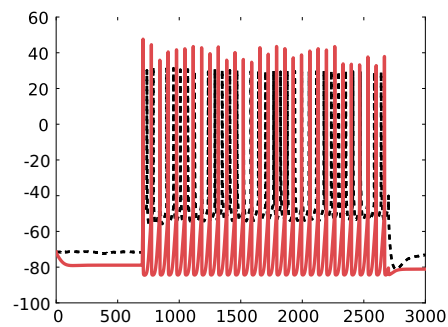
Figure 10.13: Output obtained minimizing the D_S -objective function representative of the corresponding “good runs” (the lower 10-percentile). Legend as in Figure 10.12.



(a) Overall raster plot.



(b) First out-of-sample time window.



(c) Second out-of-sample time window.

Figure 10.14: Output obtained minimizing the $D_{q=15}^{\text{Ross}}$ -objective function representative of the corresponding “good runs” (the lower 10-percentile of the $D_S(\text{out1})$ performance score). Legend as in Figure 10.12.

it appears that the values of the parameters estimates are rather consistent across the three cost functions, and the resulting ranges are relatively small compared to the search intervals of each parameters (see Table 10.7).

Computational cost of the PSO minimization procedure

As far as the computational load is concerned, the three metrics require essentially the same execution time. In fact, running the neuron model is much more time-consuming than computing any of the the spike-train distances. To give a reference of the overall computational load, consider that running ten optimisation procedures in parallel on the ISTBI computing cluster⁶ took about 5 h 57 min 42 s \pm 1 h 12 min 20 s (such statistics is computed considering the wall time of all optimization results reported here). Then, ignoring possible slowdowns due to the usage of the shared facility by other users and supposing a uniform distribution among all runs, we can estimate the mean execution time of a single PSO run to be about 35 min 46 s \pm 7 min 14 s, which is comparable to the average execution time of the ($N = 100$)-sized EnKF in the first twin experiment on the BBP model (cfr. Section 9.3).

10.5 Discussion and conclusions

In this last experiment, we aimed at assimilating Blue Brain Project experimental data produced by step input currents (see Figure 10.1a and Figure 10.2) into the detailed neuron model used by the BBP to reproduce the data for layer 2 and 3 pyramidal cells. The objective was, on the one hand to test the methods we applied using simulated data in a completely experimental setting, and on the other hand to check the quality of the BBP model by verifying its ability to exactly reproduce part of the same experimental data they used.

In sharp contrast to the twin experiments carried out in the previous chapters, we were not able to successfully apply the ensemble Kalman filter even though we adopted some earlier identified precautions (e.g. enforcing positivity constraints) and some new ones too (such as exploring a bounded search space whose size decreased attempt after attempt). Point-wise assimilation of the membrane potential trace was not possible even using a maximum-likelihood-like approach or any brute-force minimization of other cost functions based on ℓ^p -point-wise estimation of the experimental recording. Only resorting to neuron-tailored objective functions resulting from spike-train metrics allowed us to identify a suitable set of model parameters able to predict some features of the real neuron response to out-of-sample stimuli. In particular, we used a

⁶the cluster specifics are those reported at the beginning of Section 9.1

validation dataset (the experimental data in the first out-of-sample time window) to select the successful runs of the minimization procedure, and then evaluated the quality of the DA method using a different test set (the experimental data in the second out-of-sample time window). However, we first had to substantially reduce the parameter search-space size to make the minimization procedure work, but the size-reduction procedure is far from being solid and can difficultly be proposed as a standard for future studies.

Nevertheless, renouncing to point-wise estimation in such way, we were able to obtain some parameter estimates that produce reasonable out-of-sample predictions of the experimental spike trains. In particular, let us highlight that

- i)* we found both the prediction quality and the parameter estimates to be rather stable with respect to the minimized spike-train metric (either the parameter dependent Victor-Purpura SPIKE distance and van Rossum distance or the parameter-free SPIKE distance);
- ii)* the parameter estimates we identified do make the model match and predict the experimental spike trains significantly better than the BBP value θ_{BBP} (for instance, compare the red trace in Figure 10.5 to the one in Figure 10.8b).

However, let us discuss the results we obtained in relation to the prior model and the type of experimental data used for the assimilation. To begin with, what we found seems to highlight that the model proposed by the BBP is not able to point-wisely reproduce the data they assimilated. This is testified, for instance, by the fact that the traces which have a good agreement with the assimilated spike trains, all undergo a small but persistent hyperpolarization at the beginning of all time windows (see Figure 10.8, Figure 10.13b-c, and Figure 10.14b-c). It appears that when the initial I_{hyp} current is injected, the experimental traces are already at their resting state, whereas the model always hyperpolarizes in order to reach its resting potential, although both receive the same input current. In the BBP paper [149], it is reported that the leak conductance \bar{g}_L “was manually fixed to a value that created a resting potential [...] in accordance with reported values” for pyramidal cell models (see the Supplementary material at the section “Neuronal physiology”), but even considering such parameter as an unknown, our optimization procedure did not yield any parameter configuration compatible with the experimental data. This fact can also explain the EnKF failure: since such method has to track down even the first values of the experimental potential time course, it is immediately driven away from the good state-variables and parameters values which agree with the experimental data.

However, we do not think this is a proof that detailed single-neuron models cannot be fine-tuned to membrane potential data using Bayesian DA methods in general, but only that data produced by step-current **IDrest** protocols are not informative enough. Indeed, there are plenty of examples where this task has been successfully carried out: see for instance [199, 122, 206, 164, 106]. However, all these works do assimilate neural responses to much more complicated and dynamically diverse stimuli. In particular, it has been argued that input currents should drive rapid changes in the neuronal activity and explore wide dynamical ranges in order to allow parameter estimation through single-trace fitting [88].

On the other hand, the BBP fitting procedure does not aim at point-wisely fitting any potential trace produced by the **IDrest** protocol at all. In fact, the targets of such multi-objective optimization are the statistics of several electrophysiological features (resting potential, spike rate, action potential amplitudes etc.). Note that the features are extracted from the responses of different layer 5 pyramidal cells to step-currents of different amplitude, and the statistics are built aggregating the traces with similar current-amplitude to rheobase ratio. This is done so that a given neuron model accounts for the electrophysiological diversity of neural populations but, as a result, any of the inputs that generated their dataset elicits the same response as in the experimental data. Nevertheless, paying the price of needing several potential traces and aggregating them, the BBP procedure appears to get the best out of the step-current data, to enforce more constraints than point-wise fitting does, and to rely on those electrophysiological criteria that allow an experimentalist to classify given neuron types.

Summarizing, we think that the compatibility of the model and the quality of data we employed is the origin of the unsatisfactory results we obtained. If only experimental step-current data are available, the BBP fitting procedure is probably the best choice to fine-tune detailed neuron models, whereas a simpler model may be considered to achieve better point-wise fitting through DA. However, assimilating experimental recordings generated by more complicated input currents one should still be able to reproduce and predict the neural response to new stimuli employing Bayesian DA methods on realistic models as the BBP one.

To conclude, we think that carefully selecting a prior model, ensuring the dynamical richness of the data, and investigating the compatibility of the two before performing DA is of paramount importance. In particular, in view of future assimilation projects on neural networks models, one should first verify that, on the one hand, the model is able to exhibit various dynamical behaviours (e.g., neural synchronization, reproduce some characteristic oscil-

lation rates such as alpha waves, gamma oscillations, beta activity, etc.), and on the other hand that the assimilated data are rich enough to explore several network states.

List of Figures

1.1	Diagram of a general state-space model.	15
1.2	DAG of the Gaussian nonlinear state-space model described in Example 1.3	18
2.1	minAone: evolution of 100 action minimums for the Lorenz96 Gaussian problem as a function of β	39
2.2	Mindmap of data assimilation methods	41
3.1	Graphical representation of the prediction-analysis filtering update.	44
3.3	Graphical representation of the Kalman filter.	47
4.2	Graphical representation of the Kalman filter in case of bounded variables.	72
4.4	Graph of the logit function and its inverse.	75
4.5	Graphical representation of a logit-normally distributed random variable.	75
5.1	Graphs of sample sigmoid-shaped asymptotic value functions and Gaussian-like characteristic-time functions.	93
5.2	Ionic currents dynamics for different values of the membrane potential.	95
5.5	Sample trajectory of the toy model of single neuron.	99
5.6	Representation of model L23_PC_cADpyr229_1 for layer 2 and 3 pyramidal cells.	101
5.8	L23_PC_cADpyr229_1 twin experiments: in-sample simulated dataset.	108
5.9	L23_PC_cADpyr229_1 twin experiments: in-sample and both out-of-sample simulated dataset.	109
5.10	Sample solution of model (5.16) for different values of constant input current I_{ext} all with the same initial condition $(V_0, a_0)^T = (60 \text{ mV}, 0.5)^T$	112

5.11	Bifurcation diagram of the two-dimensional model (5.16) with respect to parameter $I_{\text{ext}} \in [-10 \mu\text{A cm}^{-2}, 45 \mu\text{A cm}^{-2}]$	113
6.1	Summary of the morphological types considered in the neocortical microcircuit.	121
6.2	Examples of different e-types and me-types electrical activities, and the me-types distribution across layers.	122
6.5	Sample time course of a single-exponential synapse $s(t)$	125
6.6	Sample LIF network dynamics	129
7.1	Representation of the basic operations needed for the cost-based spike train distances.	133
7.2	Van Rossum spike train distance and Schreiber et al. similarity measure.	135
7.3	Illustration of the local quantities needed to calculate the instantaneous dissimilarity values for ISI- and SPIKE-distances.	137
8.1	Toy neuron model twin experiment: sample filtering distribution.	149
8.2	Toy neuron model twin experiment: true parameter values versus filtering mean for EnKF, BF, and OPT-SIRS.	150
8.3	Toy neuron model twin experiment: parameter component of the filtering error for EnKF, BF, and OPT-SIRS.	151
8.4	Toy neuron model twin experiment: graphical representation of the definition of $\hat{\theta}$ (8.1), for component \bar{q}_L	152
8.7	Toy neuron model twin experiment: forecast skill in the first 100 ms of the generalization time window of EnKF, BF, and OPT-SIRS.	155
9.1	Unbiased twin experiment on model L23_PC_cADpyr229_1: approximated filtering distribution computed by a single run of the EnKF.	164
9.2	Unbiased twin experiment on model L23_PC_cADpyr229_1: approximated filtering distribution computed by the EnKF: detail.	165
9.3	Unbiased twin experiment on model L23_PC_cADpyr229_1: sample EnKF signal estimation performance.	167
9.4	Unbiased twin experiment on model L23_PC_cADpyr229_1: box plots of the performance scores.	170
9.5	Unbiased twin experiment on model L23_PC_cADpyr229_1: Tukey's HSD test for the ensemble size effect.	171
9.6	Unbiased twin experiment on model L23_PC_cADpyr229_1: error plot of the factorized performance score.	172

9.8	Graphical representation of the computational wall time as a function of the ensemble size N	173
9.10	Biased twin experiment on model L23_PC_cADpyr229_1: estimated p.d.f. of the D_S -performance score.	175
9.11	Biased twin experiment on model L23_PC_cADpyr229_1: raster plots of some notable runs.	175
9.13	Biased twin experiment on model L23_PC_cADpyr229_1: filtering distribution of the parameters for some notable runs.	179
9.16	Response of the fitted model in the second out-of-sample time window out2 for the “good runs” selected in the validation step.	182
10.1	Assimilation of experimental data in the BBP model: sample content of the BBP experimental dataset.	189
10.2	Assimilation of experimental data in the BBP model: experimental dataset for assimilation	191
10.4	Assimilation of experimental data in the BBP model: graphs of the univariate d_{SSE} -fit functions.	195
10.5	Assimilation of experimental data in the BBP model: numerical solutions of model L23_PC_cADpyr229_1 for different values of the modelling parameter θ	196
10.6	Assimilation of experimental data in the BBP model: graphs of the univariate D_S -fit functions.	197
10.8	Assimilation of experimental data in the BBP model: median result for two van Rossum objective functions.	203
10.9	Assimilation of experimental data in the BBP model: box-plot comparison of the in-sample results obtained minimizing the van Rossum and the Victor-Purpura objective functions.	204
10.10	Assimilation of experimental data in the BBP model: empirical p.d.f.’s of the $D_{q=15}^{SPIKE}$ results, the D_S results, and the $D_{q=15}^{Ross}$ results	206
10.11	Assimilation of experimental data in the BBP model: empirical p.d.f.’s of the $D_{q=15}^{SPIKE}$ results, the D_S results, and the $D_{q=15}^{Ross}$ results	207
10.12	Assimilation of experimental data in the BBP model: output obtained minimizing the $D_{q=15}^{SPIKE}$ -objective function representative of the corresponding “good runs” (the lower 10-percentile of the $D_S(out1)$ performance score).	209
10.13	Assimilation of experimental data in the BBP model: output obtained minimizing the D_S -objective function representative of the corresponding “good runs” (the lower 10-percentile of the $D_S(out1)$ performance score).	210

10.14	Assimilation of experimental data in the BBP model:	
	output obtained minimizing the $D_{q=15}^{\text{Ross}}$ -objective function representative of the corresponding “good runs” (the lower 10-percentile).	211

List of Tables

4.1	Summary of parameter estimation methods in DA.	59
4.3	Transformations considered for bounded variables and moments of the log-normal distribution	73
5.3	Model L23_PC_cADpyr229_1: list of ionic currents, state variables, and modelling parameters.	97
5.4	Model L23_PC_cADpyr229_1: true parameter values and unit measure	98
5.7	Model L23_PC_cADpyr229_1: parameter values.	105
6.4	Frequency of layer 2 and 3 me-types in both the in-layer and in-microcircuit composition.	123
7.4	Spike-train metrics summary	140
8.5	Toy neuron model twin experiment: mean and standard deviation of estimated parameters, with respect to the 100 in- dependent runs of each filter.	153
8.6	Toy neuron model twin experiment: mean of parameter estimation relative error in the toy model twin experiment. . . .	154
8.8	Toy neuron model twin experiment: mean L^1 -error in gen- eralization and prediction time windows and relative estimation error	156
9.7	Mean and standard deviation of the computational wall time (in hours) for different ensemble sizes.	173
9.9	Biased twin experiment on model L23_PC_cADpyr229_1: voltage estimation performance statistics and notable runs per- formance.	174
9.12	Biased twin experiment on model L23_PC_cADpyr229_1: parameter estimation performance statistics.	177
9.14	Biased twin experiment on model L23_PC_cADpyr229_1: parameter performance of some notable runs.	180

9.15	Biased twin experiment on model L23_PC_cADpyr229_1: parameter improvement rate for some notable runs.	181
10.3	Assimilation of experimental data in the BBP model: search parameter intervals for the optimiser considered in the Blue Brain Project	193
10.7	Assimilation of experimental data in the BBP model: updated search parameter intervals $I_{\text{new}}^{(k)}$	199
10.15	Mean and coefficient of variation of the parameter values un- derlying the “good runs”.	208

List of Algorithms

Algorithm 3.2	Kalman filter (KF)	46
Algorithm 3.4	Ensemble Kalman filter (EnKF)	49
Algorithm 3.5	Bootstrap filter (BF)	51
Algorithm 3.6	Optimal importance resampling (OPT-SIRS)	53

List of Abbreviations

Abbreviations

3DVAR	three dimensional variational method	KF	Kalman filter
4DVAR	four dimensional variational method	KS	Kalman smoother
ANN	artificial neural network	LIF	leaky integrate-and-fire
BF	bootstrap filter	MAP	maximum a posteriori
BM	Bayesian method	MCMC	Markov chain Monte Carlo
BRAIN	Brain Research through Advancing Innovative Neurotechnologies	MH	Metropolis-Hastings
DA	data assimilation	ML	maximum likelihood
DAG	directed acyclic graph	ODE	ordinary differential equation
EM	expectation-maximization	OPT-SIRS	optimal sequential importance resampling
EnKF	ensemble Kalman filter	PF	particle filters
EPFL	École Polytechnique Fédérale de Lausanne	PMMH	particle marginal Metropolis-Hastings
ExKF	extended Kalman filter	PSO	particle swarm optimisation
HSD	honest significant difference (statistical test)	SMC	sequential Monte Carlo methods
ISI	inter-spike interval	SSE	sum of squared errors
ISTBI	Institute for Science and Technology for Brain-Inspired Intelligence	SSM	state-space model
		UKF	unscented Kalman filter
		UPF	unscented particle filter

BBP shorthands

BBP	Blue Brain Project
BP	bipolar cell
BTC	bitufted cell
cAC	continuous accommodating
cAD	continuous adaptive
ChC	chandelier cell
DBC	double bouquet cell
e-type	electrical type
HBP	Human Brain Project
L23	layer 2 and 3
LBC	large basket cell
m-type	electrical type
MC	Martinotti cell
me-type	morpho-electrical type
NBC	nest basket cell
NGC	neurogliaform cell
NMC	neocortical microcircuit col- laboration

PC	pyramidal cell
SBC	small basket cell
SP	star pyramidal cell
TTPC1	thick-tufted pyramidal cell

Standard shorthands

a.e.	almost everywhere
c.d.f.	cumulative distribution function
cfr.	<i>confer</i> (Latin for “com- pare”)
e.g.	<i>exempli gratia</i> (Latin for “for example”)
etc.	<i>et cetera</i> (Latin for “and so forth”)
i.e.	<i>id est</i> (Latin for “that is”)
i.i.d.	independent identically- distributed
l.h.s.	left hand side
p.d.f.	probability density function
r.h.s.	right hand side
r.v.	random variable

Index

- 4DVAR, 34
- 3DVAR, 48

- acceptance density, 29
- action, 37
- analysis step, 44
- aperiodic, 26
- augmented signal variable, 60
- augmented state-space model, 60

- Bayesian methods, 59
- Bayesian quality assessment, 77
- Blue Brain Project, 87
- bootstrap filter, 51
- burn-in, 33

- Carathéodory conditions, 114
- Carathéodory existence theorem, 115
- conditional mean, 34
- consistent, 54
- convexity, 36
- corner, 138
- curse of dimensionality, **33**, 200

- data, 15
- data assimilation, 13
- data assimilation time window, **19**, 106
- derivative-based algorithms, 36
- detailed balance, 29
- deterministic discrete-time map, 17
- deterministic observation operator, 17
- directed acyclic graph, 16
- dynamical noise process, 16

- electrical type, 119

- ensemble, 50
- ensemble size, 50
- ergodic, 27
- ergodic theorem, 27
- evolutionary algorithms, 36
- existence, 54
- extended Kalman filter, 47
- extended solution, 115

- filtering distribution, 43
- first out-of-sample time window, 78
- forecast skill, 78

- generalization time window, 154
- genetic algorithms, 36

- Haas-White, 135
- hidden Markov model, 16
- hyperparameter, 63

- importance sampling distribution, 50
- importance weight, 50
- in-sample, 107
- in-sample time window, **78**, 154
- independent samplers, 31
- innovation, 46
- invariant distribution, 26
- irreducibility, 26
- ISI-distance, 137

- Kalman gain, 46
- Kalman smoother, 35

- leaky integrate-and-fire, 120
- likelihood function, 22

Lipschitz constant, 55
 Lipschitz continuous, 55

Markov chain Monte Carlo, 26
 maximum a posteriori, 34
 maximum likelihood, 34
 maximum likelihood methods, 59
 measurement absolute error, 79
 measurement noise process, 17
 Metropolis algorithms, 32
 Metropolis-Hastings, 29
 Monte Carlo approximation, 28, **50**
 morpho-electrical type, 120
 morphological type, 119

negative-log likelihood, 22
 negative-log prior, 22
 negative-log smoothing density, 23
 neocortical microcircuit collaboration, 94
 Neuron, 89, **100**

observation process, 15
 off-line, 25
 on-line, 25
 optimal sequential importance resampling, 52
 out-of-sample, 107
 out-of-sample time window, 155

particle filters, 50
 particle marginal Metropolis-Hastings, 60
 particle swarm optimisation, **36**, 202
 path integral, 37
 path space, 37
 predicted distribution, 44
 prediction step, 43
 prediction time window, 155
 prior density, 21
 prior model, 67
 proposal density, 29

proposal distribution, 50
 proposed ensemble, 50

quantile function, 64

recurrent, 26
 relative measurement error, 80

Schreiber et al., 135
 second out-of-sample time window, **78**
 section, 100
 segments, 100
 sequential Monte Carlo methods, 50
 sigma points, 48
 signal absolute error, 79
 signal estimation quality assessment, 77
 signal process, 13
 smoothing density, 23
 smoothing problem, 21
 SPIKE synchronization, 139
 SPIKE-distance, 138
 SPIKE-distance objective function, 201
 stability, 54
 standard ergodic assumptions, 26
 state-space model, 13

thinning, 33
 transition kernel, 26
 true discrete trajectory, 106
 true trajectory, 98

univariate D_S -fit function, 196
 univariate d_{SSE} -fit function, 194
 unscented Kalman filter, 48
 unscented particle filter, 54

van Rossum distance, 134
 van Rossum objective function, 201
 variational methods, 33
 Victor-Purpura inter-spike interval distance, 133
 Victor-Purpura objective function, 201
 Victor-Purpura SPIKE metric, 132

Bibliography

- [1] <https://www.humanbrainproject.eu/en/follow-hbp/news/worlds-brain-initiatives-move-forward-together/>; <https://www.brainalliance.org.au/learn/media-releases/worlds-brain-initiatives-move-forward-together/>. Declaration of Intent to Create an International Brain Initiative (IBI). December 2017.
- [2] <https://github.com/yejingxin/minAone>. Reference webpage for minAone. Last access: October 2017.
- [3] <https://projects.coin-or.org/IPOPT>. Reference webpage for the interior point optimizer IPOPT. Last access: January 2014.
- [4] <https://www-sigproc.eng.cam.ac.uk/smc/>. Webpage of Signal Processing and Communications Group at the University of Cambridge. Last access: February 2018.
- [5] <https://neuron.yale.edu/neuron/>. Reference webpage for the NEURON simulation environment. Last access: January 2017.
- [6] <https://bbp.epfl.ch/nmc-portal>. Blue Brain Project neocortical collaboration portal. Last access: April 2018.
- [7] <https://mariomulansky.github.io/PySpike/>. Reference webpage for the Python library for the numerical analysis of spike train similarity (pyspike Python module). Last access: May 2017.
- [8] <https://neuralensemble.org/ElPhAnT/>. Reference webpage for the electro-physiology analysis toolkit (elephant Python module). Last access: March 2018.
- [9] <https://github.com/BlueBrain/BluePyOpt/>. Reference webpage for the Blue Brain Python optimisation library (bluepyopt Python module). Last access: April 2017.
- [10] <https://esa.github.io/pygmo/>. Reference webpages for the Python library for bio-inspired and evolutionary optimization algorithms (pygmo Python module).

- [11] S. I. Aanonsen et al. “The ensemble Kalman filter in reservoir engineering—a review”. In: *Spe Journal* 14.03 (2009), pp. 393–412.
- [12] H. D. Abarbanel. “Predicting the Future: Completing Models of Observed Complex Systems”. In: *AMC* 10 (2013), p. 12.
- [13] L. F. Abbott. “Lapicque’s introduction of the integrate-and-fire model neuron (1907)”. In: *Brain research bulletin* 50.5-6 (1999), pp. 303–304.
- [14] R. Adams, D. Brown, and A. Constanti. “M-currents and other potassium currents in bullfrog sympathetic neurones”. In: *The Journal of Physiology* 330.1 (1982), pp. 537–572.
- [15] R. P. Agarwal and V. Lakshmikantham. *Uniqueness and nonuniqueness criteria for ordinary differential equations*. Vol. 6. World Scientific Publishing Company, 1993.
- [16] J. T. Alander. *An indexed bibliography of genetic algorithms: Years 1957-1993*. Art of CAD, 1994.
- [17] J. L. Anderson. “An adaptive covariance inflation error correction algorithm for ensemble filters”. In: *Tellus A* 59.2 (2007), pp. 210–224.
- [18] D. Aronov et al. “Neural coding of spatial phase in V1 of the macaque monkey”. In: *Journal of neurophysiology* 89.6 (2003), pp. 3304–3327.
- [19] R. B. Avery and D. Johnston. “Multiple channel types contribute to the low-voltage-activated calcium current in hippocampal CA3 pyramidal neurons”. In: *The Journal of neuroscience* 16.18 (1996), pp. 5567–5582.
- [20] V. A. Bavdekar et al. “Constrained dual ensemble Kalman filter for state and parameter estimation”. In: *2013 American Control Conference*. IEEE, 2013, pp. 3093–3098.
- [21] R. Bellman. *Adaptive control processes: a guided tour*. Princeton University Press, 1961.
- [22] L. Bengtsson, M. Ghil, and E. Källén. *Dynamic meteorology: data assimilation methods*. Vol. 36. Springer, 1981.
- [23] D. Bianchi et al. “On the mechanisms underlying the depolarization block in the spiking dynamics of CA1 pyramidal neurons”. In: *Journal of computational neuroscience* 33.2 (2012), pp. 207–225.
- [24] G. D. Birkhoff. “Proof of the ergodic theorem”. In: *Proceedings of the National Academy of Sciences* 17.12 (1931), pp. 656–660.
- [25] F. Biscani, D. Izzo, and M. Märten. *esa/pagmo2: pagmo 2.7*. doi.org/10.5281/zenodo.1217831. 2018.

- [26] M. Bocquet et al. “Data assimilation in atmospheric chemistry models: current status and future prospects for coupled chemistry meteorology models”. In: *Atmospheric chemistry and physics* 15.10 (2015), pp. 5325–5358.
- [27] Z. Botev. “The normal law under linear restrictions: simulation and estimation via minimax tilting”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 79.1 (2017), pp. 125–148.
- [28] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.
- [29] N. Brunel and M. C. Van Rossum. “Quantitative investigations of electrical nerve excitation treated as polarization”. In: *Biological Cybernetics* 97.5-6 (2007), pp. 341–349.
- [30] N. Brunel and X.-J. Wang. “Effects of neuromodulation in a cortical network model of object working memory dominated by recurrent inhibition”. In: *Journal of computational neuroscience* 11.1 (2001), pp. 63–85.
- [31] H. L. Bryant and J. P. Segundo. “Spike initiation by transmembrane current: a white-noise analysis.” In: *The Journal of physiology* 260.2 (1976), pp. 279–314.
- [32] G. Burgers, P. Jan van Leeuwen, and G. Evensen. “Analysis scheme in the ensemble Kalman filter”. In: *Monthly weather review* 126.6 (1998), pp. 1719–1724.
- [33] A. N. Burkitt. “A review of the integrate-and-fire neuron model: I. Homogeneous synaptic input”. In: *Biological cybernetics* 95.1 (2006), pp. 1–19.
- [34] W. H. Calvin and C. F. Stevens. “Synaptic noise as a source of variability in the interval between action potentials”. In: *Science* 155.3764 (1967), pp. 842–844.
- [35] C. Canuto and A. Tabacco. *Analisi matematica II: Teoria ed esercizi con complementi in rete*. Springer Science & Business Media, 2008.
- [36] O. Cappé, S. J. Godsill, and E. Moulines. “An overview of existing methods and recent advances in sequential Monte Carlo”. In: *Proceedings of the IEEE* 95.5 (2007), pp. 899–924.
- [37] N. T. Carnevale and M. L. Hines. *The NEURON book*. Cambridge University Press, 2006.

- [38] G. A. Carpenter. “A geometric approach to singular perturbation problems with applications to nerve impulse equations”. In: *Journal of Differential Equations* 23.3 (1977), pp. 335–367.
- [39] J. A. Carton and B. S. Giese. “A reanalysis of ocean climate using Simple Ocean Data Assimilation (SODA)”. In: *Monthly Weather Review* 136.8 (2008), pp. 2999–3017.
- [40] M. Çetin and S. Beyhan. “Adaptive Stabilization of Uncertain Cortex Dynamics under Joint Estimates and Input Constraints”. In: *IEEE Transactions on Circuits and Systems II: Express Briefs* (2018).
- [41] S. B. Chitrlekha et al. “A comparison of simultaneous state and parameter estimation schemes for a continuous fermentor reactor”. In: *Journal of Process Control* 20.8 (2010), pp. 934–943.
- [42] N. Chopin, P. E. Jacob, and O. Papaspiliopoulos. “SMC2: an efficient algorithm for sequential analysis of state space models”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 75.3 (2013), pp. 397–426.
- [43] K. N. Chueh, C. C. Conley, and J. A. Smoller. “Positively invariant regions for systems of nonlinear diffusion equations”. In: *Indiana University Mathematics Journal* 26.2 (1977), pp. 373–392.
- [44] C. M. Colbert and E. Pan. “Ion channel properties underlying axonal action potential initiation in pyramidal neurons”. In: *Nature neuroscience* 5.6 (2002), pp. 533–538.
- [45] D. Crisan and A. Doucet. “A survey of convergence results on particle filtering methods for practitioners”. In: *IEEE Transactions on signal processing* 50.3 (2002), pp. 736–746.
- [46] J. Cronin. *Mathematical aspects of Hodgkin-Huxley neural theory*. Vol. 7. Cambridge University Press, 1987.
- [47] V. Cutsuridis, S. Cobb, and B. P. Graham. “Encoding and retrieval in a model of the hippocampal CA1 microcircuit”. In: *Hippocampus* 20.3 (2010), pp. 423–446.
- [48] E. D’Angelo et al. “Theta-frequency bursting and resonance in cerebellar granule cells: experimental evidence and modeling of a slow K⁺-dependent mechanism”. In: *Journal of Neuroscience* 21.3 (2001), pp. 759–770.

- [49] G. Deco, E. T. Rolls, and B. Horwitz. ““What” and “where” in visual working memory: a computational neurodynamical perspective for integrating fMRI and single-neuron data”. In: *Journal of Cognitive Neuroscience* 16.4 (2004), pp. 683–701.
- [50] P. Del Moral and A. Guionnet. “On the stability of interacting processes with applications to filtering and genetic algorithms”. In:
- [51] A. Destexhe, Z. F. Mainen, and T. J. Sejnowski. “Synthesis of models for excitable membranes, synaptic transmission and neuromodulation using a common kinetic formalism”. In: *Journal of computational neuroscience* 1.3 (1994), pp. 195–230.
- [52] A. Destexhe et al. “A model of spindle rhythmicity in the isolated thalamic reticular nucleus”. In: *Journal of neurophysiology* 72.2 (1994), pp. 803–818.
- [53] K. Diba, C. Koch, and I. Segev. “Spike propagation in dendrites with stochastic ion channels”. In: *Journal of Computational Neuroscience* 20.1 (2006), pp. 77–84.
- [54] S. Ditlevsen and P. Greenwood. “The Morris–Lecar neuron model embeds a leaky integrate-and-fire model”. In: *Journal of Mathematical Biology* 67.2 (2013), pp. 239–259.
- [55] S. Ditlevsen and A. Samson. “Parameter estimation in the stochastic Morris-Lecar neuronal model with particle filter methods”. (unpublished). June 2012. URL: <https://hal.archives-ouvertes.fr/hal-00712331>.
- [56] S. Ditlevsen, A. Samson, et al. “Estimation in the partially observed stochastic Morris–Lecar neuronal model with particle filter and stochastic approximation methods”. In: *The annals of applied statistics* 8.2 (2014), pp. 674–702.
- [57] S. Diwakar et al. “Axonal Na⁺ channels ensure fast spike activation and back-propagation in cerebellar granule cells”. In: *Journal of neurophysiology* 101.2 (2009), pp. 519–532.
- [58] A. Doucet, N. De Freitas, and N. Gordon. *Sequential Monte Carlo Methods in Practice. Series Statistics For Engineering and Information Science*. 2001.
- [59] A. Doucet, S. Godsill, and C. Andrieu. “On sequential Monte Carlo sampling methods for Bayesian filtering”. In: *Statistics and computing* 10.3 (2000), pp. 197–208.

- [60] S. Druckmann et al. “A novel multiple objective optimization framework for constraining conductance-based neuron models by experimental data”. In: *Frontiers in neuroscience* 1 (2007), p. 1.
- [61] S. Druckmann et al. “Evaluating automated parameter constraining procedures of neuron models by experimental and surrogate data”. In: *Biological cybernetics* 99.4-5 (2008), p. 371.
- [62] R. C. Eberhart, J. Kennedy, et al. “A new optimizer using particle swarm theory”. In: *Proceedings of the sixth international symposium on micro machine and human science*. Vol. 1. New York, NY. 1995, pp. 39–43.
- [63] H. Eibern and H. Schmidt. “A four-dimensional variational chemistry data assimilation scheme for Eulerian chemistry transport modeling”. In: *Journal of Geophysical Research: Atmospheres* 104.D15 (1999), pp. 18583–18598.
- [64] A. P. Engelbrecht. *Computational intelligence: an introduction*. John Wiley & Sons, 2007.
- [65] B. Ermentrout. *Simulating, analyzing, and animating dynamical systems: a guide to XPPAUT for researchers and students*. Vol. 14. Siam, 2002.
- [66] G. B. Ermentrout and D. H. Terman. *Mathematical foundations of neuroscience*. Vol. 35. Springer Science & Business Media, 2010.
- [67] G. Evensen. “Inverse methods and data assimilation in nonlinear ocean models”. In: *Physica D: Nonlinear Phenomena* 77.1-3 (1994), pp. 108–129.
- [68] G. Evensen. “Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics”. In: *Journal of Geophysical Research: Oceans* 99.C5 (1994), pp. 10143–10162.
- [69] G. Evensen. *Data assimilation: the ensemble Kalman filter*. Springer Science & Business Media, 2009.
- [70] A. A. Faisal, L. P. Selen, and D. M. Wolpert. “Noise in the nervous system”. In: *Nature reviews neuroscience* 9.4 (2008), p. 292.
- [71] A. Filippov. *Differential Equations with Discontinuous Righthand Sides: Control Systems*. Vol. 18. Springer Science & Business Media, 1988.
- [72] R. FitzHugh. “Impulses and physiological states in theoretical models of nerve membrane”. In: *Biophysical journal* 1.6 (1961), pp. 445–466.
- [73] S. Gasparini and M. Migliore. “Action Potential Backpropagation”. In: *Encyclopedia of Computational Neuroscience* (2015), pp. 133–137.

- [74] N. Geir et al. “Reservoir monitoring and continuous model updating using ensemble Kalman filter”. In: *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers. 2003.
- [75] M. Ghil and P. Malanotte-Rizzoli. “Data assimilation in meteorology and oceanography”. In: *Advances in geophysics*. Vol. 33. Elsevier, 1991, pp. 141–266.
- [76] S. Gillijns et al. “What is the ensemble Kalman filter and how well does it work?” In: *American Control Conference, 2006*. IEEE. 2006, 6–pp.
- [77] J. H. Goldwyn and E. Shea-Brown. “The what and where of adding channel noise to the Hodgkin-Huxley equations”. In: *PLoS computational biology* 7.11 (2011), e1002247.
- [78] N. J. Gordon, D. J. Salmond, and A. F. Smith. “Novel approach to nonlinear/non-Gaussian Bayesian state estimation”. In: *IEE Proceedings F (Radar and Signal Processing)*. Vol. 140. 2. IET. 1993, pp. 107–113.
- [79] G. A. Gottwald and A. Majda. “A mechanism for catastrophic filter divergence in data assimilation for sparse observation networks”. In: *Nonlinear Processes in Geophysics* 20.5 (2013), pp. 705–712.
- [80] N. W. Gouwens et al. “Systematic generation of biophysically detailed models for diverse cortical neuron types”. In: *Nature communications* 9.1 (2018), p. 710.
- [81] W. Govaerts, Y. A. Kuznetsov, and A. Dhooge. “Numerical continuation of bifurcations of limit cycles in MATLAB”. In: *SIAM journal on scientific computing* 27.1 (2005), pp. 231–252.
- [82] J. S. Haas and J. A. White. “Frequency selectivity of layer II stellate cells in the medial entorhinal cortex”. In: *Journal of neurophysiology* 88.5 (2002), pp. 2422–2429.
- [83] M. Hajós et al. “Modulation of septo-hippocampal θ activity by GABA A receptors: An experimental and computational approach”. In: *Neuroscience* 126.3 (2004), pp. 599–610.
- [84] X. Han and X. Li. “An evaluation of the nonlinear/non-Gaussian filters for the sequential data assimilation”. In: *Remote Sensing of Environment* 112.4 (2008), pp. 1434–1449.
- [85] E. Hay et al. “Models of neocortical layer 5b pyramidal cells capturing a wide range of dendritic and perisomatic active properties”. In: *PLoS computational biology* 7.7 (2011), e1002107.

- [86] M. L. Hines and N. T. Carnevale. “The NEURON simulation environment”. In: *Neural computation* 9.6 (1997), pp. 1179–1209.
- [87] M. L. Hines and N. T. Carnevale. “Expanding NEURON’s repertoire of mechanisms with NMODL”. In: *Neural Computation* 12.5 (2000), pp. 995–1007.
- [88] K. H. Hobbs and S. L. Hooper. “Using complicated, wide dynamic range driving to develop models of single neurons in single recording sessions”. In: *Journal of neurophysiology* 99.4 (2008), pp. 1871–1883.
- [89] A. L. Hodgkin and A. F. Huxley. “A quantitative description of membrane current and its application to conduction and excitation in nerve”. In: *The Journal of physiology* 117.4 (1952), p. 500.
- [90] P. D. Hoff. *A first course in Bayesian statistical methods*. Springer Science & Business Media, 2009.
- [91] I. Hoteit, D.-T. Pham, and J. Blum. “A simplified reduced order Kalman filtering and application to altimetric data assimilation in Tropical Pacific”. In: *Journal of Marine systems* 36.1-2 (2002), pp. 101–127.
- [92] I. Hoteit et al. “A new approximate solution of the optimal nonlinear filter for data assimilation in meteorology and oceanography”. In: *Monthly Weather Review* 136.1 (2008), pp. 317–334.
- [93] C. Houghton and K. Sen. “A new multineuron spike train metric”. In: *Neural computation* 20.6 (2008), pp. 1495–1511.
- [94] P. R. Houser, G. De Lannoy, and J. P. Walker. *Hydrological Data Assimilation*. InTech, 2012.
- [95] P. R. Houser et al. “Integration of soil moisture remote sensing and hydrologic modeling using data assimilation”. In: *Water Resources Research* 34.12 (1998), pp. 3405–3420.
- [96] Q. J. Huys, M. B. Ahrens, and L. Paninski. “Efficient estimation of detailed single-neuron models”. In: *Journal of neurophysiology* 96.2 (2006), pp. 872–890.
- [97] Q. J. Huys and L. Paninski. “Smoothing of, and parameter estimation from, noisy biophysical recordings”. In: *PLoS Comput Biol* 5.5 (2009), e1000379.
- [98] K. Ide and C. Jones. “Special issue on mathematics of data assimilation”. In: *Physica D: Nonlinear Phenomena* 230.1–2 (2007), vii:viii. ISSN: 0167-2789. DOI: doi.org/10.1016/j.physd.2007.04.001.

- [99] E. M. Izhikevich. *Dynamical systems in neuroscience: The Geometry of Excitability and Bursting*. Computational Neuroscience. Cambridge, Massachusetts: MIT press, 2007.
- [100] E. M. Izhikevich and B. Ermentrout. “Phase model”. In: *Scholarpedia* 3.10 (2008), p. 1487. DOI: doi.org/10.4249/scholarpedia.1487.
- [101] N. Johnson, S. Kotz, and N Balakrishnan. *Continuous Univariate Probability Distributions,(Vol. 1)*. 1994.
- [102] M. I. Jordan. *An introduction to probabilistic graphical models*. Available at <http://www.cs.berkeley.edu/jordan/prelims/>. 2003.
- [103] S. J. Julier and J. K. Uhlmann. *A general method for approximating nonlinear transformations of probability distributions*. Tech. rep. Technical report, Robotics Research Group, Department of Engineering Science, University of Oxford, 1996.
- [104] S. J. Julier and J. K. Uhlmann. “New extension of the Kalman filter to nonlinear systems”. In: *Signal processing, sensor fusion, and target recognition VI*. Vol. 3068. International Society for Optics and Photonics. 1997, pp. 182–194.
- [105] S. I. Kabanikhin. “Definitions and examples of inverse and ill-posed problems”. In: *Journal of Inverse and Ill-Posed Problems* 16.4 (2008), pp. 317–357.
- [106] N. Kadakia et al. “Nonlinear statistical data assimilation for HVC-RA neurons in the avian song system”. In: *Biological cybernetics* 110.6 (2016), pp. 417–434.
- [107] R. E. Kalman. “A new approach to linear filtering and prediction problems”. In: *Journal of basic Engineering* 82.1 (1960), pp. 35–45.
- [108] E. Kalnay. *Atmospheric modeling, data assimilation and predictability*. Cambridge university press, 2003.
- [109] E. Kalnay. *Atmospheric modeling, data assimilation and predictability*. Cambridge university press, 2003.
- [110] N. Kantas et al. “An overview of sequential Monte Carlo methods for parameter estimation in general state-space models”. In: *IFAC Proceedings Volumes* 42.10 (2009), pp. 774–785.
- [111] N. Kantas et al. “On particle methods for parameter estimation in state-space models”. In: *Statistical science* 30.3 (2015), pp. 328–351.
- [112] K. Katterbauer, I. Hoteit, and S. Sun. “EMSE: Synergizing EM and seismic data attributes for enhanced forecasts of reservoirs”. In: *Journal of Petroleum Science and Engineering* 122 (2014), pp. 396–410.

- [113] J. P. Keener and J. Sneyd. *Mathematical physiology*. 2nd. Springer, New York, NY, 2009. DOI: doi.org/10.1007/978-0-387-79388-7.
- [114] D. Kelly, K. Law, and A. M. Stuart. “Well-posedness and accuracy of the ensemble Kalman filter in discrete and continuous time”. In: *Nonlinearity* 27.10 (2014), p. 2579.
- [115] C. Keppenne et al. “Ensemble Kalman filter assimilation of temperature and altimeter data with bias correction and application to seasonal prediction”. In: *Nonlinear processes in Geophysics* 12.4 (2005), pp. 491–503.
- [116] G. Kitagawa. “A self-organizing state-space model”. In: *Journal of the American Statistical Association* (1998), pp. 1203–1215.
- [117] P. Kloeden and E. Platen. *Numerical solution of stochastic differential equations*. Springer-Verlag New York, 1992.
- [118] M Kohler et al. “Small-conductance, calcium-activated potassium channels from mammalian brain”. In: *Science* 273.5282 (1996), p. 1709.
- [119] M. Kohn. “Computer modeling at the national biomedical simulation resource”. In: *Computers & Mathematics with Applications* 18.10-11 (1989), pp. 919–924.
- [120] M. H. Kole, S. Hallermann, and G. J. Stuart. “Single Ih channels in pyramidal neuron dendrites: properties, distribution, and impact on action potential output”. In: *The Journal of neuroscience* 26.6 (2006), pp. 1677–1687.
- [121] A. Korngreen and B. Sakmann. “Voltage-gated K⁺ channels in layer 5 neocortical pyramidal neurones from young rats: subtypes and gradients”. In: *The Journal of Physiology* 525.3 (2000), pp. 621–639.
- [122] M. Kostuk et al. “Dynamical estimation of neuron and network properties II: path integral Monte Carlo methods”. In: *Biological cybernetics* 106.3 (2012), pp. 155–167.
- [123] J. H. Kotecha and P. M. Djuric. “Gaussian particle filtering”. In: *IEEE Transactions on Signal Processing* 51.10 (2003), pp. 2592–2601. ISSN: 1053-587X. DOI: doi.org/10.1109/TSP.2003.816758.
- [124] T. Kreuz. “Measures of spike train synchrony”. In: 6.10 (2011), p. 11934. DOI: doi.org/10.4249/scholarpedia.11934.
- [125] T. Kreuz, M. Mulansky, and N. Bozanic. “SPIKY: A graphical user interface for monitoring spike train synchrony”. In: *Journal of neurophysiology* 113.9 (2015), pp. 3432–3445.

- [126] T. Kreuz et al. “Measuring spike train synchrony and reliability”. In: *BMC Neuroscience* 8.2 (2007), p. 1.
- [127] T. Kreuz et al. “Measuring multiple spike train synchrony”. In: *Journal of neuroscience methods* 183.2 (2009), pp. 287–299.
- [128] T. Kreuz et al. “Time-resolved and time-scale adaptive measures of spike train synchrony”. In: *Journal of neuroscience methods* 195.1 (2011), pp. 92–106.
- [129] T. Kreuz et al. “Monitoring spike train synchrony”. In: *Journal of neurophysiology* 109.5 (2013), pp. 1457–1472.
- [130] M. Krysta et al. “Data assimilation for short-range dispersion of radionuclides: An application to wind tunnel data”. In: *Atmospheric Environment* 40.38 (2006), pp. 7267–7279.
- [131] Y. A. Kuznetsov. *Elements of applied bifurcation theory*. Vol. 112. Springer Science & Business Media, 2013.
- [132] S Lange, A. Lupascu, et al. “Data-driven computational modeling of CA1 hippocampal principal cells and interneurons”. In: *BMC Neuroscience* 18.Suppl (2017), P177.
- [133] L. Lapique. “Recherches quantitatives sur l’excitation électrique des nerfs traitée comme une polarisation”. In: *Journal de physiologie et de pathologie générale* 9 (1907), pp. 620–635.
- [134] K. Law, A. Stuart, and K. Zygalakis. *Data Assimilation*. Vol. 62. Texts in Applied Mathematics. Springer, 2015.
- [135] K. J. Law and A. M. Stuart. “Evaluating data assimilation algorithms”. In: *Monthly Weather Review* 140.11 (2012), pp. 3757–3782.
- [136] F. Le Gland, V. Monbet, and V.-D. Tran. “Large sample asymptotics for the ensemble Kalman filter”. PhD thesis. INRIA, 2009.
- [137] S. Li, J. Li, and Z. Li. “An improved unscented kalman filter based decoder for cortical brain-machine interfaces”. In: *Frontiers in neuroscience* 10 (2016), p. 587.
- [138] J. Liepe et al. “A framework for parameter estimation and model selection from experimental data in systems biology using approximate Bayesian computation”. In: *Nature protocols* 9.2 (2014), pp. 439–456.
- [139] H. Lindén et al. “Modeling the spatial reach of the LFP”. In: *Neuron* (2011). ISSN: 08966273. DOI: 10.1016/j.neuron.2011.11.006.
- [140] H. F. Lopes and R. S. Tsay. “Particle filters and Bayesian inference in financial econometrics”. In: *Journal of Forecasting* 30.1 (2011), pp. 168–209.

- [141] A. López-Cuevas et al. “State and parameter estimation of a neural mass model from electrophysiological signals during the status epilepticus”. In: *NeuroImage* 113 (2015), pp. 374–386.
- [142] A. Lorenc et al. “The Met. Office global three-dimensional variational data assimilation scheme”. In: *Quarterly Journal of the Royal Meteorological Society* 126.570 (2000), pp. 2991–3012.
- [143] A. C. Lorenc. “Analysis methods for numerical weather prediction”. In: *Quarterly Journal of the Royal Meteorological Society* 112.474 (1986), pp. 1177–1194.
- [144] A. C. Lorenc and T. Payne. “4D-Var and the butterfly effect: Statistical four-dimensional data assimilation for a wide range of scales”. In: *Quarterly Journal of the Royal Meteorological Society* 133.624 (2007), pp. 607–614.
- [145] W. W. Lytton. “Optimizing synaptic conductance calculation for network simulations”. In: *Neural computation* 8.3 (1996), pp. 501–509.
- [146] J. Magistretti and A. Alonso. “Biophysical properties and slow voltage-dependent inactivation of a sustained sodium current in entorhinal cortex layer-II principal neurons a whole-cell and single-channel study”. In: *The Journal of general physiology* 114.4 (1999), pp. 491–509.
- [147] Z. F. Mainen and T. J. Sejnowski. “Reliability of spike timing in neocortical neurons”. In: *Science* 268.5216 (1995), pp. 1503–1506.
- [148] M. Markaki, S. Orphanoudakis, and P. Poirazi. “Modelling reduced excitability in aged CA1 neurons as a calcium-dependent process”. In: *Neurocomputing* 65 (2005), pp. 305–314.
- [149] H. Markram et al. “Reconstruction and simulation of neocortical microcircuitry”. In: *Cell* 163.2 (2015), pp. 456–492.
- [150] P. Z. Marmarelis and K.-I. Naka. “White-noise analysis of a neuron chain: an application of the Wiener theory”. In: *Science* 175.4027 (1972), pp. 1276–1278.
- [151] S. Masoli et al. “Single neuron optimization as a basis for accurate biophysical modeling: the case of cerebellar granule cells”. In: *Frontiers in cellular neuroscience* 11 (2017), p. 71.
- [152] A. Mazzoni. “From Single Neuron Activity to Network Information Processing: Simulating Cortical Local Field Potentials and Thalamus Dynamic Regimes with Integrate-and-Fire Neurons”. In: *Mathematical and Theoretical Neuroscience*. Springer, 2017, pp. 1–23.

- [153] H. M. Menegaz et al. “A systematization of the unscented Kalman filter theory”. In: *IEEE Transactions on automatic control* 60.10 (2015), pp. 2583–2598.
- [154] R van der Merwe et al. *The unscented particle filter Technical Report CUED*. Tech. rep. F-INFENG/TR 380, Cambridge University Engineering Department, Cambridge, England, 2000.
- [155] N. Metropolis et al. “Equation of state calculations by fast computing machines”. In: *The journal of chemical physics* 21.6 (1953), pp. 1087–1092.
- [156] S. P. Meyn and R. L. Tweedie. *Markov chains and stochastic stability*. 2nd ed. Springer Science & Business Media, 2012.
- [157] M. Migliore et al. “Distributed organization of a brain microcircuit analyzed by three-dimensional modeling: the olfactory bulb”. In: *Frontiers in computational neuroscience* 8 (2014), p. 50.
- [158] H. Moradkhani et al. “Uncertainty assessment of hydrologic model states and parameters: Sequential data assimilation using the particle filter”. In: *Water resources research* 41.5 (2005).
- [159] C. Morris and H. Lecar. “Voltage oscillations in the barnacle giant muscle fiber”. In: *Biophysical journal* 35.1 (1981), pp. 193–213.
- [160] M. Mulansky and T. Kreuz. “PySpike-A Python library for analyzing spike train synchrony”. In: *arXiv preprint arXiv:1603.03293* (2016).
- [161] P. Munõz-Gutiérrez and E Giraldo. “Ensemble Kalman filter for state estimation of brain activity by considering a large scale nonlinear dynamical model”. In: *VII Latin American Congress on Biomedical Engineering CLAIB 2016, Bucaramanga, Santander, Colombia, October 26th-28th, 2016*. Springer. 2017, pp. 445–448.
- [162] J. Nagumo, S. Arimoto, and S. Yoshizawa. “An active pulse transmission line simulating nerve axon”. In: *Proceedings of the IRE* 50.10 (1962), pp. 2061–2070.
- [163] J. Nocedal and S. Wright. *Numerical optimization*. Springer Science & Business Media, 2006.
- [164] A. Nogaret et al. “Automatic construction of predictive neuron models through large scale assimilation of electrophysiological data”. In: *Scientific reports* 6 (2016), p. 32749.
- [165] D. S. Oliver, A. C. Reynolds, and N. Liu. *Inverse theory for petroleum reservoir characterization and history matching*. Cambridge University Press, 2008.

- [166] D. T. Pham, J. Verron, and M. C. Roubaud. “A singular evolutive extended Kalman filter for data assimilation in oceanography”. In: *Journal of Marine systems* 16.3-4 (1998), pp. 323–340.
- [167] P. Poirazi, T. Brannon, and B. W. Mel. “Pyramidal neuron as two-layer neural network”. In: *Neuron* 37.6 (2003), pp. 989–999.
- [168] R. Poli, J. Kennedy, and T. Blackwell. “Particle swarm optimization”. In: *Swarm intelligence* 1.1 (2007), pp. 33–57.
- [169] N. Politi, J. Feng, and W. Lu. “Comparing data assimilation filters for parameter estimation in a neuron model”. In: *Neural Networks (IJCNN), 2016 International Joint Conference on*. IEEE, 2016.
- [170] J Prakash, S. C. Patwardhan, and S. L. Shah. “Constrained nonlinear state estimation using ensemble Kalman filters”. In: *Industrial & Engineering Chemistry Research* 49.5 (2010), pp. 2242–2253.
- [171] A. Quarteroni, R. Sacco, and F. Saleri. *Numerical mathematics*. Vol. 37. Springer Science & Business Media, 2010.
- [172] R. Q. Quiroga, T. Kreuz, and P. Grassberger. “Event synchronization: a simple and fast method to measure synchronicity and time delay patterns”. In: *Physical review E* 66.4 (2002), p. 041904.
- [173] S. Ramaswamy et al. “The neocortical microcircuit collaboration portal: a resource for rat somatosensory cortex”. In: *Frontiers in neural circuits* 9 (2015).
- [174] P. Rebeschini, R. Van Handel, et al. “Can local particle filters beat the curse of dimensionality?” In: *The Annals of Applied Probability* 25.5 (2015), pp. 2809–2866.
- [175] R. H. Reichle. “Data assimilation methods in the Earth sciences”. In: *Advances in Water Resources* 31.11 (2008), pp. 1411–1418.
- [176] R. H. Reichle, D. B. McLaughlin, and D. Entekhabi. “Hydrologic data assimilation with the ensemble Kalman filter”. In: *Monthly Weather Review* 130.1 (2002), pp. 103–114.
- [177] J Rettig et al. “Characterization of a Shaw-related potassium channel family in rat brain.” In: *The EMBO Journal* 11.7 (1992), p. 2473.
- [178] I Reuveni et al. “Stepwise repolarization from Ca²⁺ plateaus in neocortical pyramidal cells: evidence for nonhomogeneous distribution of HVA Ca²⁺ channels in dendrites”. In: *The Journal of neuroscience* 13.11 (1993), pp. 4609–4621.
- [179] C. P. Robert and G. Casella. *Monte Carlo statistical methods*. Springer New York, 1999.

- [180] S. Robert and H. R. Künsch. “Localizing the ensemble Kalman particle filter”. In: *Tellus A: Dynamic Meteorology and Oceanography* 69.1 (2017), p. 1282016.
- [181] M. C. W. van Rossum. “A novel spike distance”. In: *Neural Computation* 13.4 (2001), pp. 751–763.
- [182] A. Roth and M. C. van Rossum. “Modeling synapses”. In: *Computational modeling methods for neuroscientists* 6 (2009), pp. 139–160.
- [183] T. Sauer. “Numerical solution of stochastic differential equation in finance”. In: *Handbook of computational finance* (2012), pp. 529–550.
- [184] C. Schillings and A. M. Stuart. “Analysis of the ensemble Kalman filter for inverse problems”. In: *SIAM Journal on Numerical Analysis* 55.3 (2017), pp. 1264–1290.
- [185] S. Schreiber et al. “A new correlation-based measure of spike timing reliability”. In: *Neurocomputing* 52 (2003), pp. 925–931.
- [186] E. Sejdić and L. A. Lipsitz. “Necessity of noise in physiology and medicine”. In: *Computer methods and programs in biomedicine* 111.2 (2013), pp. 459–470.
- [187] S. Shalev-Shwartz and S. Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.
- [188] B. Shan et al. “UKF-based closed loop iterative learning control of epileptiform wave in a neural mass model”. In: *Cognitive neurodynamics* 9.1 (2015), pp. 31–40.
- [189] Y. Shu et al. “Selective control of cortical axonal spikes by a slowly inactivating K⁺ current”. In: *Proceedings of the National Academy of Sciences* 104.27 (2007), pp. 11453–11458.
- [190] D. Simon. “Optimal state estimation: Kalman, H infinity, and nonlinear approaches”. In: 1st ed. John Wiley & Sons, 2006. Chap. 14, pp. 433–459.
- [191] J.-A. Skjervheim et al. “Incorporating 4D seismic data in reservoir simulation models using ensemble Kalman filter”. In: *SPE journal* 12.03 (2007), pp. 282–292.
- [192] J. Smoller. *Shock waves and reaction-diffusion equations*. Vol. 258. Springer-Verlag, 1983.
- [193] C. Soto-Trevino et al. “Computational model of electrically coupled, intrinsically distinct pacemaker neurons”. In: *Journal of neurophysiology* 94.1 (2005), pp. 590–604.

- [194] M Spreng et al. “Central nervous system activation by noise”. In: *Noise and health* 2.7 (2000), p. 49.
- [195] J. Tabak, C. R. Murphey, and L. Moore. “Parameter estimation methods for single neuron models”. In: *Journal of computational neuroscience* 9.3 (2000), pp. 215–236.
- [196] O. Talagrand and P. Courtier. “Variational assimilation of meteorological observations with the adjoint vorticity equation. I: Theory”. In: *Quarterly Journal of the Royal Meteorological Society* 113.478 (1987), pp. 1311–1328.
- [197] J.-N. Thepaut and P. Courtier. “Four-dimensional variational data assimilation using the adjoint of a multilevel primitive-equation model”. In: *Quarterly Journal of the Royal Meteorological Society* 117.502 (1991), pp. 1225–1254.
- [198] B. Toth. “Python scripting for dynamical parameter estimation in IPOPT”. In: *SIAG/OPT Views-and-News* 21.1 (2010), pp. 1–8.
- [199] B. A. Toth et al. “Dynamical estimation of neuron and network properties I: variational methods”. In: *Biological cybernetics* 105.3-4 (2011), pp. 217–237.
- [200] G Triantafyllou, I Hoteit, and G Petihakis. “A singular evolutive interpolated Kalman filter for efficient data assimilation in a 3-D complex physical–biogeochemical model of the Cretan Sea”. In: *Journal of Marine Systems* 40 (2003), pp. 213–231.
- [201] R. Van Der Merwe et al. “The Unscented particle filter Technical Report CUED/F-INFENG/TR 380”. In: *Cambridge University Engineering Department* (2000).
- [202] R. Van Der Merwe et al. “The unscented particle filter”. In: *Advances in neural information processing systems*. 2001, pp. 584–590.
- [203] W. Van Geit et al. “BluePyOpt: Leveraging open source software and cloud infrastructure to optimise model parameters in neuroscience”. In: *Frontiers in Neuroinformatics* 10 (2016), p. 17. DOI: doi.org/10.3389/fninf.2016.00017.
- [204] P. J. Van Laarhoven and E. H. Aarts. *Simulated annealing: Theory and applications*. Springer, 1987.
- [205] P. J. Van Leeuwen and G. Evensen. “Data assimilation and inverse methods in terms of a probabilistic formulation”. In: *Monthly Weather Review* 124.12 (1996), pp. 2898–2913.

- [206] D. V. Vavoulis et al. “A self-organizing state-space-model approach for parameter estimation in hodgkin-huxley-type models of single neurons”. In: *PLoS Comput Biol* 8.3 (2012), e1002401.
- [207] J. D. Victor. “Spike train metrics”. In: *Current opinion in neurobiology* 15.5 (2005), pp. 585–592.
- [208] J. D. Victor. “Spike Train Distance”. In: *Encyclopedia of Computational Neuroscience*. Ed. by D. Jaeger and R. Jung. Springer, 2015, pp. 2808–2814. DOI: doi.org/10.1007/978-1-4614-6675-8_409.
- [209] J. D. Victor and K. P. Purpura. “Nature and precision of temporal coding in visual cortex: a metric-space analysis”. In: *Journal of neurophysiology* 76.2 (1996), pp. 1310–1326.
- [210] J. D. Victor and K. P. Purpura. “Metric-space analysis of spike trains: theory, algorithms and application”. In: *Network* 8 (1997), pp. 127–164.
- [211] A. Wächter and L. T. Biegler. “On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming”. In: *Mathematical programming* 106.1 (2006), pp. 25–57.
- [212] J. P. Walker and P. R. Houser. “Hydrologic data assimilation”. In: *Advances in water science methodologies. Londres: Taylor & Francis, ed 1* (2005), pp. 25–48.
- [213] J. Wang et al. “Data assimilation of membrane dynamics and channel kinetics with a neuromorphic integrated circuit”. In: *Biomedical Circuits and Systems Conference (BioCAS), 2016 IEEE*. IEEE. 2016, pp. 584–587.
- [214] X.-J. Wang and G. Buzsáki. “Gamma oscillation by synaptic inhibition in a hippocampal interneuronal network model”. In: *Journal of neuroscience* 16.20 (1996), pp. 6402–6413.
- [215] X. Wang et al. “A survey of recent advances in particle filters and remaining challenges for multitarget tracking”. In: *Sensors* 17.12 (2017), p. 2707.
- [216] L. M. Ward and P. E. Greenwood. “1/f noise”. In: *Scholarpedia* 2.12 (2007), p. 1537.
- [217] J. N. Weiss. “The Hill equation revisited: uses and misuses.” In: *The FASEB Journal* 11.11 (1997), pp. 835–841.
- [218] J. Ye et al. “Estimating the biophysical properties of neurons with intracellular calcium dynamics”. In: *Physical Review E* 89.6 (2014), p. 062714.

- [219] J. Ye et al. “Improved variational methods in statistical data assimilation”. In: *Nonlinear Processes in Geophysics* 22.2 (2015), pp. 205–213.
- [220] J. Ye et al. “Systematic variational method for statistical nonlinear state and parameter estimation”. In: *Physical Review E* 92.5 (2015), p. 052901.
- [221] M. Zafari, A. C. Reynolds, et al. “Assessing the uncertainty in reservoir description and performance predictions with the ensemble Kalman filter”. In: *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers. 2005.
- [222] Y. Zhang et al. “Real-time air quality forecasting, part I: History, techniques, and current status”. In: *Atmospheric Environment* 60 (2012), pp. 632–655.
- [223] Y. Zhang et al. “Real-time air quality forecasting, part II: State of the science, current research needs, and future prospects”. In: *Atmospheric Environment* 60 (2012), pp. 656–676.